前　言

神经网络，或者更精确地说人工神经网络，是一种植根于许多学科的技术，其中涉及神经科学、数学、统计学、物理学、计算机科学和工程学。神经网络具有的一个重要性质，即在有教师或无教师的情况下能够从输入数据中进行学习的能力，这使得它在不同领域中得到应用，如建模、时间序列分析、模式识别、信号处理和控制。

由于神经网络具有多学科性，本书对该主题进行了综合论述，并给出了大量例子、基于计算机的实验、习题以及参考文献进行补充分析。

本书由四部分组成，组织如下：

1. 介绍材料，由第1章和第2章组成。第1章大体上定性描述什么是神经网络，它们的性质、组成及其怎样和人工智能相联系。这章以一些历史注释结束。第2章提供学习过程的许多侧面的概述及其统计性质。该章引进了一个重要概念，即Vapnik-Chervonenkis(VC)维数，用于度量学习机器所实现的一簇分类函数的容量。

2. 有教师学习机器，由第3章至第7章组成。第3章研究这部分中最简单的神经网络：涉及一个或多个输出神经元但无隐藏神经元的网络。该章描述最小均方(LMS)算法(在设计线性自适应滤波器时非常流行)和感知器收敛定理。第4章给出利用反向传播算法训练的多层感知器的完全处理。这个算法(代表LMS算法的一种推广)已经作为神经网络的推进器而出现。第5章给出另一类分层神经网络即径向基函数网络详细的数学处理，它们的构成包括一层基函数。这一章强调在设计RBF网络中正则化理论的作用。第6章描述一类比较新的学习机器，即支持向量机，它的理论建立在第2章给出的统计学习理论的材料上。本书第二部分以第7章结束，讨论委员会机器，它的构成包括几个学习者作为组成部分。在这一章我们描述总体平均，推举和分层混合专家三种不同的构建委员会机器的方法。

3. 无教师学习机器，由第8章至第12章组成。第8章把Hebb学习应用到主分量分析。第9章把自组织学习的另一形式，也就是竞争学习，应用于构造以自组织映射著称的计算映射。这两章突出强调学习规则根植于神经生物学。第10章注意于设计无监督学习算法的信息理论，强调它们在建模、图像处理和独立分量分析中的应用。第11章描述植根于和信息理论有密切关系的统计力学的自监督学习机器。第12章，介绍动态规划和它与增强式学习的关系。

4. 非线性动态系统，由第13章至15章组成。第13章描述一类由短期记忆和分层前馈网络结构组成的动态系统。第14章强调涉及使用反馈的非线性动态系统所引起的稳定性问题。该章还讨论联想记忆的例子。第15章描述另一类非线性动态系统，即递归网络，它依赖于使用反馈完成输入－输出映射。

本书后记简要描述神经网络在构造用于模式识别、控制和信号处理的智能机器时所起的作用。

本书的组织在神经网络研究生课程的使用上给予了很大灵活性，教师可根据需要灵活选择讲课内容。全书中总共包括15个基于计算机的实验，其中有13个实验需使用MATLAB。

MATLAB 实验的文件可直接从以下网站下载:

ftp://ftp.mathworks.com/pub/books/haykin

http://www.mathworks.com/books/

每章后都附有习题。许多习题具有挑战性，不仅能检查本书的使用者对本书所包含的资料掌握的程度，而且扩充了这些资料。

工程师、计算机科学家和物理学家也会从本书获益。希望本书对其他学科，如心理学和神经科学的研究人员，也会有所帮助。

Simon Haykin

于 Hamilton，Ontario

1998 年 2 月

缩写和符号

缩写

AI　　artificial intelligence　人工智能
APEX　　adaptive principal components extraction　自适应主分量分析
AR　　autoregressive　自回归

BBTT　　back propagation through time　通过时间的反向传播
BM　　Boltzmann machine　Boltzmann 机
BP　　back propagation　反向传播
b/s　　bits per second　每秒比特率
BOSS　　bounded, one-sided saturation　有界，单边饱和
BSB　　brain-state-in-a-box　盒中脑状态
BSS　　Blind source (signal) separation　盲源(信号)分离

CART　　classification and regression tree　分类和回归树
cmm　　correlation matrix memory　相关矩阵记忆
CV　　cross-validation　交叉确认

DEKF　　decoupled extended Kalman filter　解耦扩展 Kalman 滤波器
DFA　　deterministic finite-state automata　确定性有限状态自动机
DSP　　digital signal processor　数字信号处理器

EKF　　extended Kalman filter　扩展 Kalman 滤波器
EM　　expectation-maximization　期望最大化

FIR　　finite-duration impulse response　有限时间冲击响应
FM　　frequency-modulated (signal)　频率调制(信号)

GEKF　　global extended Kalman filter　全局扩展 Kalman 滤波器
GCV　　generalized cross-validation　广义交叉确认
GHA　　generalized Hebbian algorithm　广义 Hebb 算法
GSLC　　generalized sidelobe canceler　广义旁瓣消除器

HME　　hierarchical mixture of expert　分层混合专家

HMM　hidden Markov model　隐 Markov 模型
Hz　hertz　赫兹

ICA　independent component analysis　独立分量分析
Infomax　maximum mutual information　最大互信息

KR　kernel regression　核回归

LMS　least-mean-square　最小均方
LR　likelihood ratio　似然比
LTP　long-term potentiation　长期电位(LPT)
LTD　lone-term depression　长期衰减
LR　likelihood ratio　似然比
LVQ　learning vector quantization　学习向量量化

MCA　minor component analysis　次分量分析
MDL　minimum description length　最小描述长度
ME　mixture of expert　混合专家
MFT　mean-field theory　平均场理论
MIMO　multiple input-multiple output　多输入多输出
ML　maximum likelihood　最大似然
MLP　multilayer perceptron　多层感知器
MRAC　model reference adaptive control　模型参考自适应控制

NARMA　nonlinear autoregressive moving average　非线性自回归滑动平均
NARX　nonlinear autoregressive with exogenous input　具有外部输入的非线性自回归
NDP　neuron-dynamic programming　神经动态规划
NW　Nadaraya-Watson (estimator)　Nadaraya-Watson(估计器)
NWKR　Nadaraya-Watson kernel regression　Nadaraya-Watson 核回归

OBD　optimal brain damage　最优脑损伤
OBS　optimal brain surgeon　最优脑外科
OCR　optical character recognition　光学字符识别
ODE　ordinary differential equation　常微分方程

PAC　probably approximately correct　可能近似正确
PCA　principal component analysis　主分量分析
pdf　probability density function　概率密度函数
pmf　probability mass function　概率质量函数

RBF	radial basis function 径向基函数
RMLP	recurrent multilayer perceptron 递归多层感知器
RTRL	real-time recurrent learning 实时递归学习
SIMO	single input-multiple output 单输入多输出
SISO	single input-single output 单输入单输出
SNR	signal-to-noise ratio 信噪比
SOM	self-organizing map 自组织映射
SRN	simple recurrent network(also referred to as Elman's recurrent network) 简单递归网络(也称为 Elman 递归网络)
SVD	singular value decomposition 奇异值分解
SVM	support vector machine 支持向量机
TDNN	time-delay neural network 时延神经网络
TLFN	time lagged feedforward network 时间滞后前馈网络
VC	Vapnik-Chervononkis (dimension) Vapnik-Chervononkis(维数)
VLSI	very-large-scale integration 超大规模集成
XOR	exclusive OR 异或

重要的符号

a	action 动作
$\mathbf{a}^T\mathbf{b}$	inner product of vectors **a** and **b** 向量 **a** 和 **b** 的内积
$\mathbf{a}\mathbf{b}^T$	output product of vectors **a** and **b** 向量 **a** 和 **b** 的外积
$\binom{l}{m}$	binomial coefficient 二项式系数
$A \cup B$	unions of A and B A 和 B 的并
B	inverse of temperature 温度的逆
b_k	bias applied to neuron k 神经元 k 的偏置
$\cos(\mathbf{a}, \mathbf{b})$	cosine of the angle between vectors **a** and **b** 向量 **a** 和 **b** 夹角的余弦
D	depth of memory 记忆深度
$D_{f\|g}$	Kullback-Leibler divergence between probability density functions f and g 概率密度函数 f 和 g 之间的 Kullback-Leibler 散度
$\tilde{\mathbf{D}}$	adjoint of operator **D** 算子 **D** 的伴随
E	energy function 能量函数
E_i	energy of state i in statistical mechanics 统计力学中状态 i 的能量
E	statistical expectation operator 统计期望算子

$\langle E \rangle$	average energy 平均能量
erf	error function 误差函数
erfc	complimentary error function 误差函数的补
exp	exponential 指数
$\mathscr{E}_{av}$	average squared error or sum of squared error 平均平方误差或平方误差和
$\mathscr{E}(n)$	instantaneous value of the sum of squared error 平方误差和的瞬时值
$\mathscr{E}_{total}$	total sum of error squares 总平方误差和
F	free energy 自由能量
$f_{\mathbf{X}}(\mathbf{X})$	probability density function of random vector **X** 随机向量 **X** 的概率密度函数
$\mathscr{F}^*$	subset (network) with the smallest minimum empirical risk 经验风险最小值最小的子集(网络)
$\mathbf{H}$	Hessian matrix Hessian 矩阵
$\mathbf{H}^{-1}$	inverse of matrix **H** 矩阵 **H** 的逆
i	square root of -1, also denoted by j -1 的平方根，亦记作 j
$\mathbf{I}$	identity matrix 单位矩阵
$\mathbf{I}$	Fisher's information matrix Fisher 信息矩阵
J	mean-square error 平均平方误差
$\mathbf{J}$	Jacobian matrix Jacobi 矩阵
$\mathbf{K}(n, n-1)$	error covariance matrix in Kalman filter theory Kalman 滤波理论中的误差协方差矩阵
$\mathbf{K}^{1/2}$	square root of matrix **K** 矩阵 **K** 的平方根
$\mathbf{K}^{T/2}$	transpose of square root of matrix **K** 矩阵 **K** 的平方根的转置
k_B	Boltzmann constant Boltzmann 常数
log	logarithm 对数
$L(\mathbf{w})$	log-likelihood function of weight vector **w** 权值向量 **w** 的对数似然函数
$\mathscr{L}(\mathbf{w})$	log-likelihood function of weight vector **w** based on a single example 单样本的权值向量 **w** 的对数似然函数
$\mathbf{M}_c$	controllability matrix 可控性矩阵
$\mathbf{M}_o$	observability matrix 可观察性矩阵
n	discrete time 离散时间
p_i	probability of state i in statistical mechanics 统计力学中状态 i 的概率
p_{ij}	transition probability from state i to state j 从状态 i 到状态 j 的转移概率
$\mathbf{P}$	stochastic matrix 随机矩阵
P_c	probability of correct classification 正确分类的概率
P_e	probability of error 误差概率
$P(e \mid \mathscr{C})$	conditional probability of error e given that the input is drawn from class $\mathscr{C}$ 从类 $\mathscr{C}$ 中输入时误差 e 的条件概率
p_α^+	probability that the visible neurons of a Boltzmann machine are in state α, given that the

network is in its clamped condition(i.e., positive phase) 假设网络处于箝制条件(即正向阶段)时，Boltzmann 机的可见神经元状态为 α 的概率

p_α^- probability that the visible neurons of a Boltzmann machine are in state α, given that the network is in its free-running condition(i.e., negative phase) 假设网络处于自由运行条件(即负向阶段)时，Boltzmann 机的可见神经元状态为 α 的概率

$\hat{r}_x(j,k;n)$ estimate of autocorrelation function of $x_j(n)$ and $x_k(n)$ $x_j(n)$和 $x_k(n)$的自相关函数估计

$\hat{r}_{dx}(k;n)$ estimate of cross-correlation function of $d(n)$ and $x_k(n)$ $d(n)$和 $x_k(n)$的交叉相关函数估计

$\mathbf{R}$ correlation matrix of an input vector 输入向量的相关矩阵

t continuous time 连续时间

T temperature 温度

$\mathcal{T}$ training set(sample) 训练集(样本)

tr trace of a matrix operator 矩阵算子的迹

var variance operator 方差算子

$V(\mathbf{x})$ Lyapunov function of state vector $\mathbf{x}$ 状态向量 $\mathbf{x}$ 的 Lyapunov 函数

v_j induced local field or activation potential of neuron j 神经元 j 的诱导局部域或激活位势

$\mathbf{w}_o$ optimum value of synaptic weight vector 突触权值向量的最优值

w_{kj} synaptic weight of synapse j belonging to neuron k 属于神经元 k 的突触 j 的突触权值

$\mathbf{w}^*$ optimum weight vector 最优权植向量

$\bar{\mathbf{x}}$ equilibrium value of state vector $\mathbf{x}$ 状态向量 $\mathbf{x}$ 的平衡值

$\langle x_j \rangle$ average of state x_j in a "thermal" sense "热"意义下状态 x_j 的平均

$\hat{x}$ estimate of x, signified by the use of a caret(hat) x 的估计，用加字符号^(帽符号)表示

$|x|$ absolute value(magnitude) of x x 的绝对值(幅度)

x^* complex conjugate of x, signified by asterisk as superscript 状态 x 的复共轭，用星号 * 作上标

$\|\mathbf{x}\|$ Euclidean norm (length) of vector $\mathbf{x}$ 向量 $\mathbf{x}$ 的欧几里德范数(长度)

$\mathbf{x}^T$ transpose of vector $\mathbf{x}$, signified by the superscript T 向量 $\mathbf{x}$ 的转置，用上标 T 表示

z^{-1} unit delay operator 单位延迟算子

Z partition function 剖分函数

$\delta_j(n)$ local gradient of neuron j at time n 神经元 j 在时刻 n 的局部梯度

Δw small change applied to weight w 权值 w 的微小改变

∇ gradient operator 梯度算子

∇^2 Laplacian operator Laplace 算子

$\nabla_w J$ gradient of J with respect to w J 关于 w 的梯度

$\nabla \cdot \mathbf{F}$	divergence of vector $\mathbf{F}$　向量 $\mathbf{F}$ 的散度
η	learning-rate parameter　学习率参数
κ	cumulant　累积量
μ	policy　策略
θ_k	threshold applied to neuron k(i.e., negative of bias b_k)　神经元 k 的阈值(即偏置 b_k 的负值)
λ	regularization parameter　正则化参数
λ_k	k th eigenvalue of a square matrix　方阵的第 k 个特征值
$\varphi_k(\cdot)$	nonlinear activation function of neuron k　神经元 k 的非线性激活函数
$\in$	symbol for "belong to"　"属于"符号
$\cup$	symbol for "union of"　"并"符号
$\cap$	symbol for "intersection of"　"交"符号
$*$	symbol for convolution　"卷积"符号
$+$	superscript symbol for pseudoinverse of a matrix　矩阵伪逆的上标符号

开区间和闭区间

- 变量 x 的开区间(a,b)表示 $a<x<b$。
- 变量 x 的闭区间$[a,b]$表示 $a\leqslant x\leqslant b$。
- 变量 x 的半闭半开区间$[a,b)$表示 $a\leqslant x<b$；类似地，变量 x 的半开半闭区间$(a,b]$表示 $a<x\leqslant b$。

最小和最大

- 符号 $\arg\min_{\mathbf{w}} f(\mathbf{w})$表示函数 $f(\mathbf{w})$关于变元向量 $\mathbf{w}$ 的最小值。
- 符号 $\arg\max_{\mathbf{w}} f(\mathbf{w})$表示函数 $f(\mathbf{w})$关于变元向量 $\mathbf{w}$ 的最大值。

目　录

第1章 导　　言

1.1　什么是神经网络

自从认识到人脑的计算与传统的数字计算机相比是完全不同的方式开始，关于人工神经网络(一般称为“神经网络”)的研究工作就开始了。人脑是一个高度复杂的、非线性的和并行的计算机器(信息处理系统)。人脑能够组织它的组成成分，即神经元，以比今天已有的最快的计算机还要快许多倍的速度进行特定的计算(如模式识别、感知和运动神经控制)。例如，考虑人类视觉，这是一个信息处理的任务(Marr, 1982; Levine, 1985; Churchland and Sejnowski, 1992)。视觉系统功能是为我们提供一个关于周围环境的表示，并且更重要的是提供我们和环境交互所需的信息。具体讲，完成一个感知识别任务(例如识别一张被嵌入陌生场景的熟悉的脸)人脑大概需要 100 ~ 200 毫秒，而一台传统的计算机却要花费几天时间才能完成一个相对简单得多的任务。

再举一个例子：考虑一只蝙蝠的声纳。声纳就是一个活动回声定位系统。除了提供目标(例如飞行的昆虫)有多远的信息外，蝙蝠的声纳可以搜集目标的相对速度、目标大小、目标不同特征的大小以及它的方位角和仰角的信息(Suga, 1990a, b)。所有信息都从目标的回声中提取，而所有需要的复杂神经计算只在李子般大小的脑中完成。事实上，一只回声定位的蝙蝠可以灵巧地以很高的成功率追逐和捕捉目标，这一点可以让雷达或声纳工程师们自叹弗如。

那么，人脑或蝙蝠的脑是如何做到这一点的呢？脑一出生就有精巧的构造和具有通过我们通常称为“经验”而建立它自己规则的能力。确实，经验是经时间积累的，人脑在出生后头两年内发生了最戏剧性的发展(即硬连接)，但是发展将超越这个阶段并继续进行。

一个“发展中”的神经元是与可塑的人脑同义的。可塑性允许一个发展中的神经系统适应它的周边环境。可塑性似乎是人脑中作为信息处理单元的神经元的功能的关键，同样，它在人工神经元组成的神经网络中亦是如此。最普通形式的神经网络就是对人脑完成特定任务或感兴趣功能的方法进行建模的机器；网络一般用电子器件实现或者用软件在数字计算机上模拟。在本书中，我们主要介绍重要的神经网络，这种网络通过学习过程来实现有用的计算。为了获得好的结果，神经网络使用一个很庞大的简单计算单元间的相互连接，这些简单计算单元称为“神经元”或者“处理单元”。据此我们给出将神经网络看作一种自适应机器的定义[1]：

一个神经网络是一个由简单处理元构成的规模宏大的并行分布式处理器。天然具有存储经验知识和使之可用的特性。神经网络在两个方面与人脑相似：

1. 神经网络获取的知识是从外界环境中学习得来的。

2. 互连神经元的连接强度，即突触权值，用于储存获取的知识。

用于完成学习过程的程序称为*学习算法*，其功能是以有序的方式改变网络的突触权值以获得想要的设计目标。

突触权值修改提供神经网络设计的传统方法。这种方法和线性自适应滤波器理论很接

近。滤波器理论已经很好地建立起来并成功应用在很多领域(Widrow and Stearns,1985;Haykin,1996)。但是神经网络修改它自身的拓扑结构亦是可能的，这也和人脑的神经元会死亡和新的突触连接会生长的情况相适应。

神经网络在文献中也称为神经计算机、连接主义网络、并行分布式处理器等。本书一律使用“神经网络”这个术语，偶尔也用“神经计算机”或“连接主义网络”。

神经网络的优点

神经网络的计算能力很明显有以下两点：(1)大规模并行分布式结构。(2)神经网络学习能力以及由此而来的泛化能力。泛化是指神经网络对不在训练(学习)集中的数据可以产生合理的输出。这两种信息处理能力让神经网络可以解决一些当前还不能处理的复杂的(大型)问题。但是在实践中，神经网络不能单独做出解答，它们需要被整合在一个协调一致的系统工程方法中。具体讲，一个复杂问题往往被分解成若干相对简单的任务，而神经网络处理与其能力相符的子任务。但是，我们在建立一个可以模拟人脑的计算机结构(如果可能)之前还有很长路要走，认识这一点是很重要的。

神经网络具有下列性质和能力：

1. 非线性。一个人工神经元可以是线性或者是非线性的。一个由非线性神经元互联而成 [2] 的神经网络自身是非线性的，并且非线性是一种分布于整个网络中的特殊性质。非线性是一个很重要的性质，特别当如果产生输入信号(如语音信号)内部的物理机制是天生非线性时。

2. 输入输出映射。有监督学习或有教师学习是一个学习的流行范例，涉及使用带标号的训练样本或任务例子对神经网络的突触权值进行修改。每个样本由一个惟一的输入信号和相应期望响应组成。从一个训练集中随机选取一个样本给网络，网络就调整它的突触权值(自由参数)，以最小化期望响应和由输入信号以适当的统计准则产生的实际响应之间的差别。使用训练集中的很多例子重复神经网络的训练，直到网络到达没有显著的突触权值修正的稳定状态为止。先前用过的例子可能还要在训练期间以不同顺序重复使用。因此对当前问题网络通过建立输入输出映射从例子中进行学习。这样一个方法使人想起了无参数统计推断的研究，它是非模型估计的统计处理的一个分支，或者从生物学角度看，称为 tabula rasa 学习(Geman et al.,1992)。这儿使用“非参数”表示的一个事实是，没有对输入数据的统计模型作任何先验假设。比如，考虑一个模式分类任务，这里的要求是把代表具体物体或事件的输入信号分类到几个预先分好的类中去。在这个问题的非参数方法中，要求利用例子集“估计”输入信号空间中模式分类任务的任意判决边界，并且不使用概率分布模型。有监督学习范例隐含了一个类似的观点，这提示神经网络的输入输出映射和非参数统计推断之间的一个相近的类比。

3. 适应性。神经网络嵌入了一个调整自身突触权值以适应外界变化的能力。特别是，一个在特定运行环境下接受训练的神经网络，对环境条件不大的变化可以容易进行重新训练。而且，当它在一个时变环境(即它的统计特性随时间变化)中运行时，网络突触权值就可以设计成随时间变化。用于模式识别、信号处理和控制的神经网络与它的自适应能力耦合，就可以变成能进行自适应模式识别、自适应信号处理和自适应控制的有效工具。作为一个一般规则，在保证系统保持稳定时一个系统的自适应性越好，当要求在一个时变环境下运行时它的性能就越具鲁棒性。但是，需要强调的是，自适应性不一定导致鲁棒性，实际可能相 [3] 反。比如，一个暂态自适应系统可能变化过快，以至对寄生干扰有反应，这将引起系统性能

的急剧恶化。为最大限度实现自适应性，系统的主要时间常数应该长到可以忽略寄生干扰，而短到可以反应环境的重要变化。这是一个*稳定性－可塑性困境*(Grossberg,1988b)。

4. *证据响应*。在模式识别的问题中，神经网络可以设计成既提供不限于选择哪一个特定模式的信息，也提供决策的*置信度*的信息。后者可以用来拒判那些出现的过于模糊的模式。有这些信息，网络的分类性能就会改善。

5. *背景的信息*。神经网络的特定结构和激发状态代表知识。网络中每一个神经元潜在地都受网络中所有其他神经元全局活动的影响。因此，背景信息自然由一个神经网络处理。

6. *容错性*。一个以硬件形式实现后的神经网络有天生容错的潜质，或者*鲁棒计算*的能力，意即它的性能在不利运行条件下逐渐下降。比如，一个神经元或它的连接损坏了，存储模式的回忆在质量上被削弱。但是，由于网络信息存储的分布特性，在网络的总体响应严重恶化之前这种损坏是分散的。因此，原则上，一个神经网络的性能显示了一个缓慢恶化而不是灾难性的失败。有一些关于鲁棒性计算的经验证据，但通常它是不可控的。为了确保网络事实上的容错性，有必要在设计训练网络的算法时采用正确的度量(Kerlirzin and Vallet,1993)。

7. *VLSI 实现*。神经网络的大规模并行性使它具有快速处理某些任务的潜在能力。这一特性使得神经网络很适合用*超大规模集成*(very-large-scale-integrated,VLSI)技术实现。VLSI 的一个特殊优点是提供一个以高度分层的方式捕捉真实复杂性行为的方法。

8. *分析和设计的一致性*。基本上，神经网络作为信息处理器具有通用性。我们这样说是在这样的意义下，即涉及神经网络的应用的所有领域都使用同样记号。这种特征以不同的方式表现出来：

- 神经元：不管形式如何，在所有的神经网络中都代表一个*相同成分*。
- 这种共性使得在不同应用中的神经网络*共享*相同的理论和学习算法成为可能。
- 模块化网络可以用模块的*无缝集成*来实现。

9. *神经生物类比*。神经网络的设计是由对人脑的类比引发的，人脑是一个容错的并行处理的活生生的例子，说明这种处理不光在物理上可实现的而且还是快速高效的。神经生物学家将(人工)神经网络看作是一个解释神经生物现象的研究工具。另一方面，工程师注意神经生物学是将其作为解决复杂问题的新思路，这些问题比基于常规的硬件线路设计技术所能解决的问题更复杂。下面两个例子说明了这两种观点： 4

- 在 Anastasio(1993)中，比较了前庭视觉反射的线性系统模型和基于在 1.6 节描述及第 15 章里详细描述的递归网络的神经网络模型。前庭视觉反射(vestibulo-ocular reflex, VOR)是眼球运动系统的一部分，其作用是让眼球向与头转动方向相反的方向运动，以维持视觉(视网膜)图像的稳定性。VOR 由前庭核酸的前端神经元调节，前端神经元从前庭感知神经元中接受头部旋转信息并处理，将结果告知眼球肌肉的动作神经元。输人(头部旋转信息)和输出(眼球旋转)可以精确确定，因此 VOR 很适合用来建模。另外，它是比较简单的反射作用 ，并且其组成神经元的神经生理学的内容已经被很好阐述。在三种神经类型中，前端神经元(反射内层神经元)在前庭神经核酸中是最复杂也是最引人注意的。VOR 以前已经用集块线性系统描述器和控制理论模型化了。这些模型对解释 VOR 的整体性质有一些作用，但是对其组成神经元特性的了解却用处不大。这种情况通过神经网络的模型已经被大大改善了。VOR 的递归网络模型(使用第 15 章描述的实时递归学习算法设计)能重现和解释调节 VOR 的神经元

的许多特性，包括处理信号时的静态、动态、非线性和分布式特性，特别是前庭核酸神经元(Anastasio,1993)。

- 视网膜不同于人脑的其他任何部分，是我们开始将外部环境的物理图像投射到一行接受器上形成的视觉表示和第一个神经图像结合的地方。它是眼球后部的神经组织薄层，其功能是将光学图像转换成神经图像并沿光神经传输给大量的视觉中枢以便进一步处理。这是一个复杂的工作，可以从视网膜的突触组织得到证明。在脊椎动物的视网膜中，光图像转化成神经图像的过程由三个阶段组成(Sterling,1990)：
(i)受体神经元层的图像传导。
(ii)结果信号(对光刺激的反应产生)由化学性突触传输给一层双极细胞。
(iii)同样，由化学性突触把结果信号传给称为神经节细胞的输出神经元。
在两个突触阶段(即从受体到双极细胞和从双极细胞到神经节细胞)，有专门侧向连接的分别称为水平细胞的神经元和无长突细胞的神经元。这些神经元的工作是修改突触层之间的传输。另外有被叫做中间网状细胞的离心元素；它们的工作是将信号从内部突触层传到外部突触层。一些研究人员已经建立了模拟视网膜结构的电子芯片(Mahowald and Mead,1989;Boahen and Ardreou,1992;Boahen,1996)。这些电子芯片称为神经形态集成电路，这个术语由 Mead(1989)所创造。一个神经形态的图像传感器由一排感光器与每个图形元素(像素)的模拟回路结合而成。它能模拟视网膜适应局部的亮度变化、检测边缘和检测运动。神经生物学模拟，例如神经形态集成电路，有另一个重要的应用：它提供一种希望和信念，并在一定程度上提供一种存在性证明，即对神经生物结构的物理上的了解对电子学工艺和超大规模集成电路技术有多方面的影响。

5

有了神经生物学的启示，我们对人脑及其组织的结构层次作简要的考察看来是合适的。

1.2 人脑

人的神经系统可看作 3 阶段系统，如同图 1-1 所描绘的框图。系统的中央是人脑，由神经网络表示，它连续地接收信息，感知它并做出适当的决定。图中有两组箭头，从左到右的箭头表示携带信息的信号通过系统向前传输，从右到左的箭头表示系统中的反馈。感受器把人体或外界环境的刺激转换成电冲击，对神经网络(大脑)传送信息。神经网络的效应器转换神经网络产生的电冲击为可识别的响应作为系统输出。

由于 Ramóny Cajál(1911)的开创性工作(他引入神经元作为人脑结构成分的思想)，理解人脑的努力已经简单多了。通常，神经元比硅逻辑门要慢 5 到 6 个数量级；硅逻辑门中的事件发生在纳秒(10^{-9}s)级，而在神经中的事件发生在毫秒(10^{-3}s)级。但是人脑由运行速度相对较慢的神经元构成，神经元(神经细胞)数目确实惊人，而且它们之间具有大量的互联。估计人的皮质有大约 100 亿神经元和大约 6 亿兆突触或连接(Shepherd and Koch,1990)。脑中的网络是高效结构。特别是，脑的能量效率每秒每个操作大约为 10^{-16}焦耳，而今天所用的最好计算机的相应值是每秒每个操作大约 10^{-6}焦耳(Faggin,1991)。

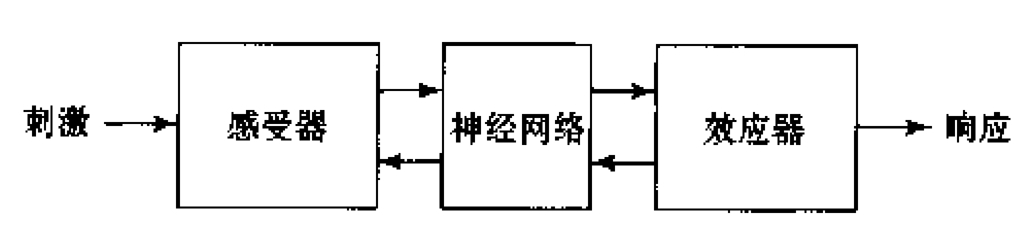


图 1-1 神经系统的框图表示

突触是调节神经元之间相互作用的

基本结构和功能单位。最普通的一类突触是化学突触，它运行如下。前突触过程释放发送器物质，扩散到神经元之间的突触连接，然后作用于后突触过程。这样突触就完成了突触前端的电信号和化学信号的转换，然后返回突触后端电信号(Shepherd and Koch,1990)。用电学术语，这样的元素称为非互逆的两端口设备。在传统的神经组织描述中，仅假设突触是一个简单的连接，能加载兴奋或抑制，但不同时作用在接受神经元。[6]

我们曾提到过，可塑性允许发展神经系统以适应周边环境(Eggermont,1990;Churchland and Sejnowski,1992)。在成年人的脑中，可塑性可以解释两个机能：创建神经元间的新连接和修改连接。轴突(即传导线路)和树突(即接受区域)组成两种细胞长纤维，它们在形态上互相区别；轴突有光华的表面，较少的分支，比较长，而树突正相反(之所以这样称呼是因为它和树相似)，它有不规则的表面和更多的分支(Freeman,1975)。脑中的不同部分有很多种形状和大小的神经元。图1-2是一种锥形细胞，它在脑皮层中是常见的。和其他许多神经元一样，它从树突刺接收大部分输入信号；可以从图1-2中看到树突片段细节。锥形细胞可以有

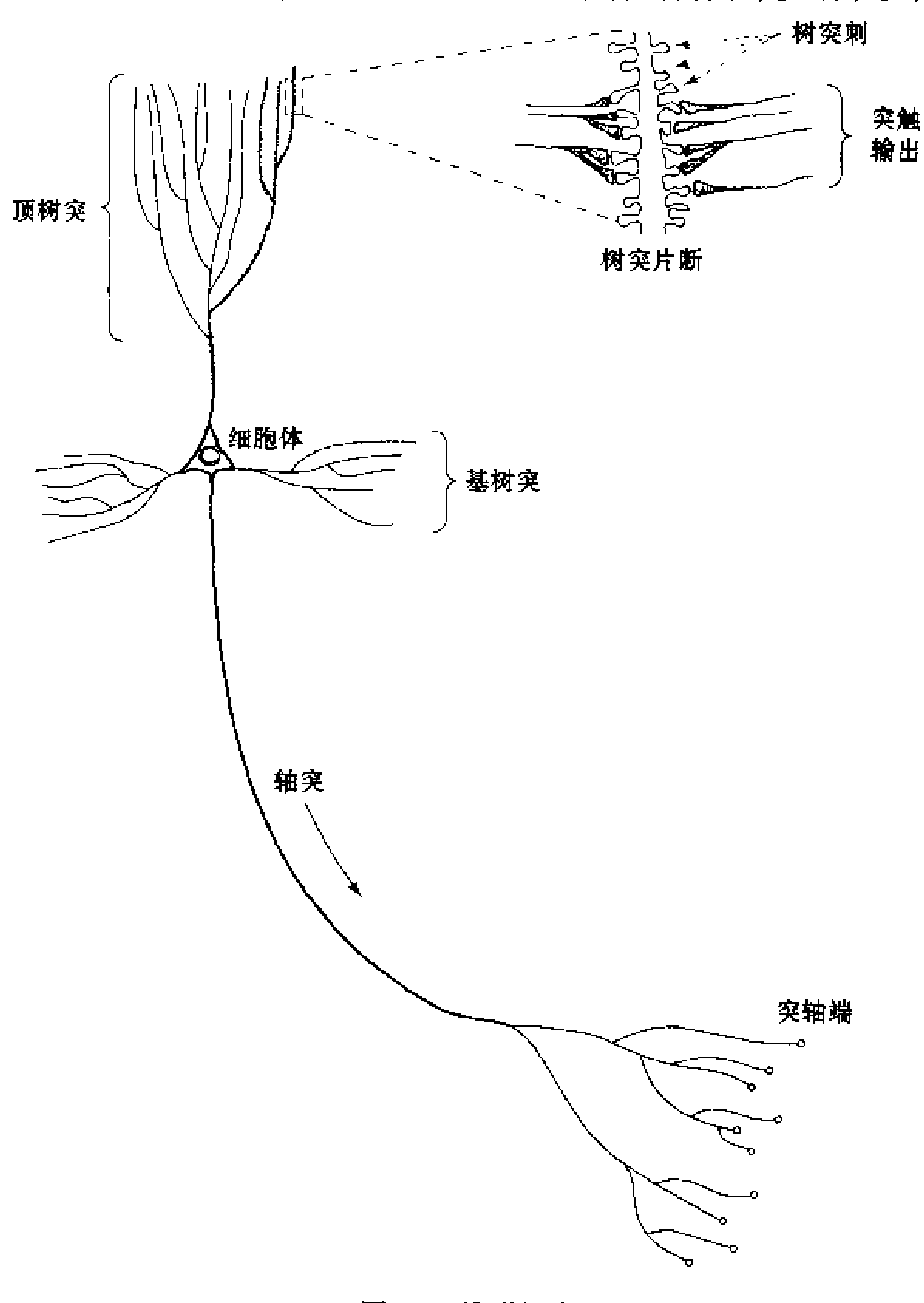


图1-2　锥形细胞

一万个或更多的突触与其他细胞连接，它可以投射到数以千计的目标细胞。

大多数神经元把它们的输出转化成一系列简短的电压脉冲编码。这些脉冲，一般称为动作电位或冲击，产生于神经元细胞体或其附近并以恒定的电压和振幅穿越个体神经元。神经元间使用动作电位通信是由轴突的物理性质决定的。轴突很长很细，有很高的电阻和非常大的电容，这二者分布于轴突中。因此可以用 RC 传输线路来建模，用"线路方程"这个术语来描述轴突中的信号传播。对传播机制的分析揭示电压在传输中随距离指数衰减，在到达另一端时会变得很小。动作电位提供了克服这个问题的方法(Anderson, 1995)。

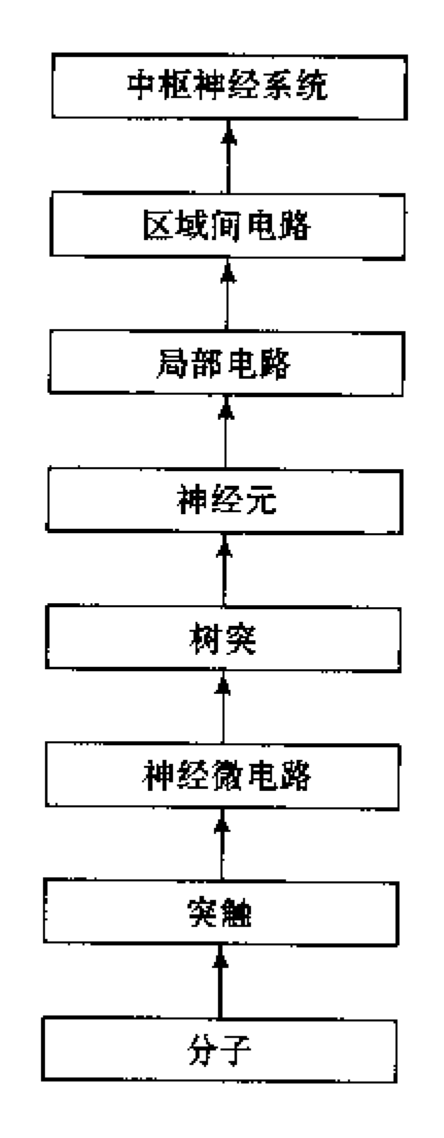


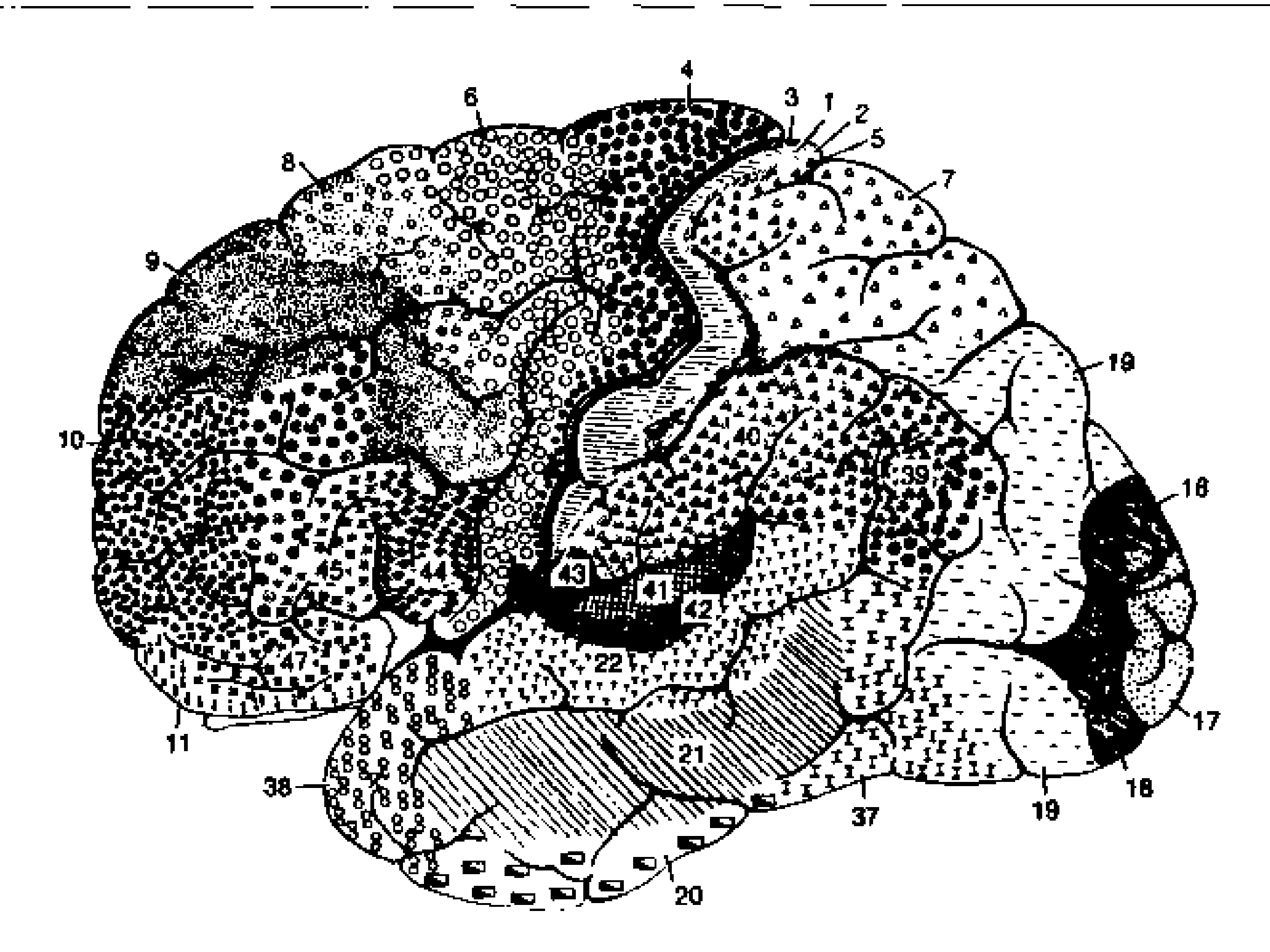
图 1-3 脑组织的分层结构

在人脑中，有大小解剖组织之分，机能也有高下之别。图 1-3 显示脑组织交织水平的层次结构，这已经在广泛的关于脑局部区域的分析工作中显现出来(Shepherd and Koch, 1990; Churchland and Sejnowski, 1992)。*突触*表示最基本的层次，其活动依赖于分子和离子。其后的层次有神经微电路、树突树和最后的神经元。*神经微电路*指突触集成，组织成可以产生感兴趣的功能操作的连接模式。它就像一个由晶体管集成的硅片，最小的尺寸用微米(μm)度量，最快的操作速度用毫秒度量，神经微电路被组织成属于神经元个体的*树突树*的树突子单元。整个*神经元*大约为 100μm 大小，包含几个树突子单元。*局部电路*(大约 1mm 大小)处在其次的复杂性水平，由具有相似或不同性质的神经元组成，这些神经元集成完成脑局部区域的特征操作。再次为*区域间电路*，由通路、柱子和局部解剖图组成，牵涉脑中不同部分的多个区域。

*局部解剖图*被组织成响应输人感知器信息。它们经常被组织成片束状，如同在上丘中一样。上丘中视觉、听觉和人体触觉区以层邻接的方式放置，使得空间中相应点的刺激处于各层的下面或上面。图 1-4 表示由 Brodmann(Brodal, 1981)做出的大脑皮质的细胞结构图。它清楚表明不同的感觉信息(运动、触觉、视觉、听觉等)被有序地映射到大脑皮层的相应位置。在复杂性的最后一级，局部解剖图和其他的区域间电路成为*中央神经系统*传递特定行为的媒介。

认识到在这里描绘的结构分层组织是人脑的独有特征非常重要。我们在数字计算机中找不到这种结构，在人工神经网络中也无法近似地重构它们。但是，我们仍在向图 1-3 中描述的类似的分级计算的层状结构缓慢推进。用以构造的神经网络的人工神经元和人脑中的神经元相比确实比较初级。我们目前能设计的网络和人脑中初级的局部电路和区域间电路相当。但是，真正令人满意的是过去 20 年间我们在许多前沿有了显著进步。以神经生物类比作为灵感的源泉，加上我们具有的理论和技术工具的这些财富，下一个十年我们对人工网络的理解一定会更加深人。

本书的主要兴趣限于从工程学角度研究人工神经网[2]。我们从描述人工神经元模型开始研究神经网，神经元模型是本书后面各章讨论神经网络的基础。



不同区域由它们的层厚度及其内部细胞类型标示。一些最重要的特殊区域如下
运动皮质：运动区，区域4；前运动区，区域6；前端眼球区，区域8。人体触觉皮质：区域3，1，2，视觉皮质：区域17，18，19。听觉皮质：区域41，42(摘自A.Brodal，1981；经Oxford University Press允许)

图1-4 大脑皮质细胞结构图

1.3 神经元模型

神经元是神经网络操作的基本信息处理单位。方框图1-5显示神经元的模型，它是(人工)神经网络的设计基础。我们在这里给出神经元模型的三种基本元素：

1. *突触或连接链*，每一个都由其权值或者强度作为特征。特别是，在连到神经元 k 的突触 j 上的输入信号 x_j 被乘以 k 的突触权重 w_{kj}。注意突触权值 w_{kj} 的下标的写法很重要。第一个下标指查询神经元，第二个下标指权值所在的突触的输入端。和人脑中的突触不一样，人工神经元的突触权值有一个范围，可以取正值也可以取负值。

2. *加法器*，用于求输入信号被神经元的相应突触加权的和。这个操作构成一个线性组合器。

3. *激活函数*，用来限制神经元输出振幅。激活函数也称为*压制函数*，由于它将输出信号压制(限制)到允许范围之内的一定值。通常，一个神经元输出的正常幅度范围可写成单位闭区间[0,1]或者另一种区间[-1,+1]。

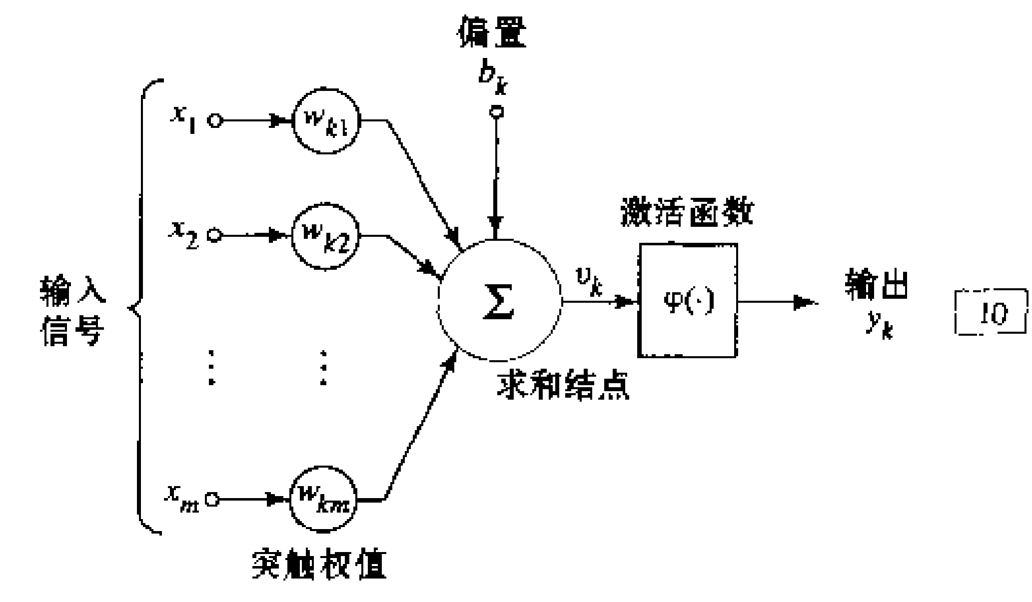


图1-5 神经元的非线性模型

[10]

图1-5的神经元模型也包括一个外部*偏置*，记为 b_k。偏置的作用是根据其为正或为负，相应地增加或降低激活函数的网络输入。

用数学术语，我们可以用如下一对方程描述一个神经元 k：

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1.1}$$

$$y_k = \varphi(u_k + b_k) \tag{1.2}$$

其中$x_1, x_2, \cdots, x_m$ 是输入信号，$w_{k1}, w_{k2}, \cdots, w_{km}$是神经元$k$ 的突触权值，u_k 是输入信号的线性组合器的输出，偏置为b_k，激活函数为$\varphi(\cdot)$，y_k是神经元输出信号。偏置b_k的作用是对图 1-5 模型中的线性组合器的输出 u_k 作仿射变换，如下所示：

$$v_k = u_k + b_k \tag{1.3}$$

特别地，根据偏置 b_k 取正或取负，神经元 k 的诱导局部域或激活电位 v_k 和线性组合器输出 u_k 的关系如图 1-6 所示；以后我们将使用"诱导局部域"这个术语。注意到由于这个仿射变换的作用，v_k 与 u_k 的图形不再经过原点。

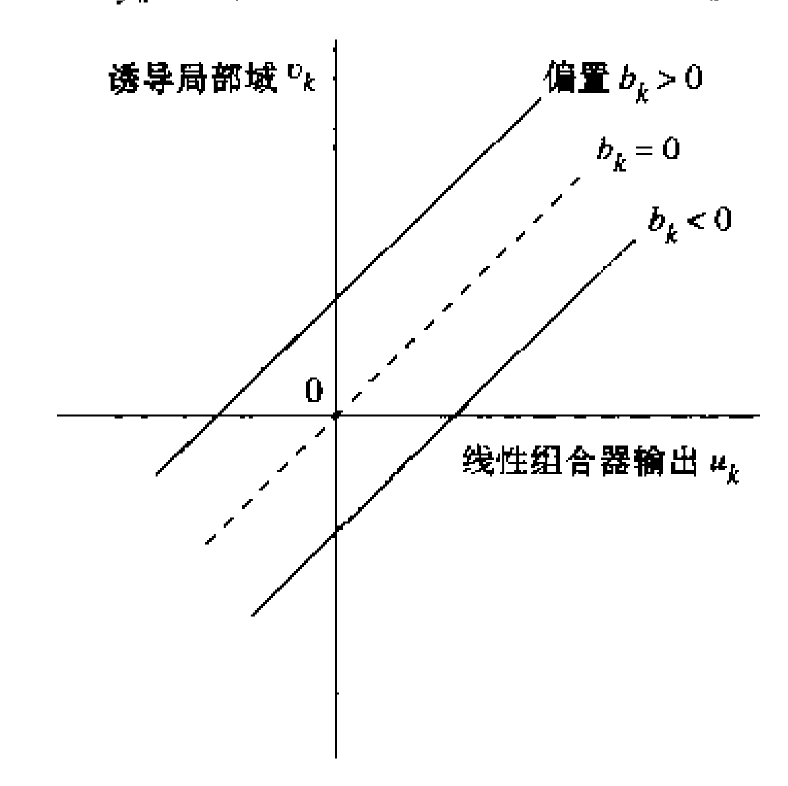


图 1-6 偏置产生的仿射变换
（注意 $u_k=0$ 时 $v_k=b_k$）

偏置 b_k 是人工神经元 k 的外部参数。我们可以像在方程(1.2)中一样考虑它。同样，我们可以结合方程(1.1)和(1.3)得到如下公式：

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \tag{1.4}$$

$$y_k = \varphi(v_k) \tag{1.5}$$

11

在(1.4)中，我们加上一个新的突触，其输入是

$$x_0 = +1 \tag{1.6}$$

权值是

$$w_{k0} = b_k \tag{1.7}$$

我们因此得到了神经元 k 的新模型图 1-7。在这个图中，偏置的作用是做两件事：(1)添加新的固定输入 +1；(2)添加新的等于偏置 b_k 的突触权值。虽然形式上图 1-5 和图 1-7 的模型不相同，但在数学上它们是等价的。

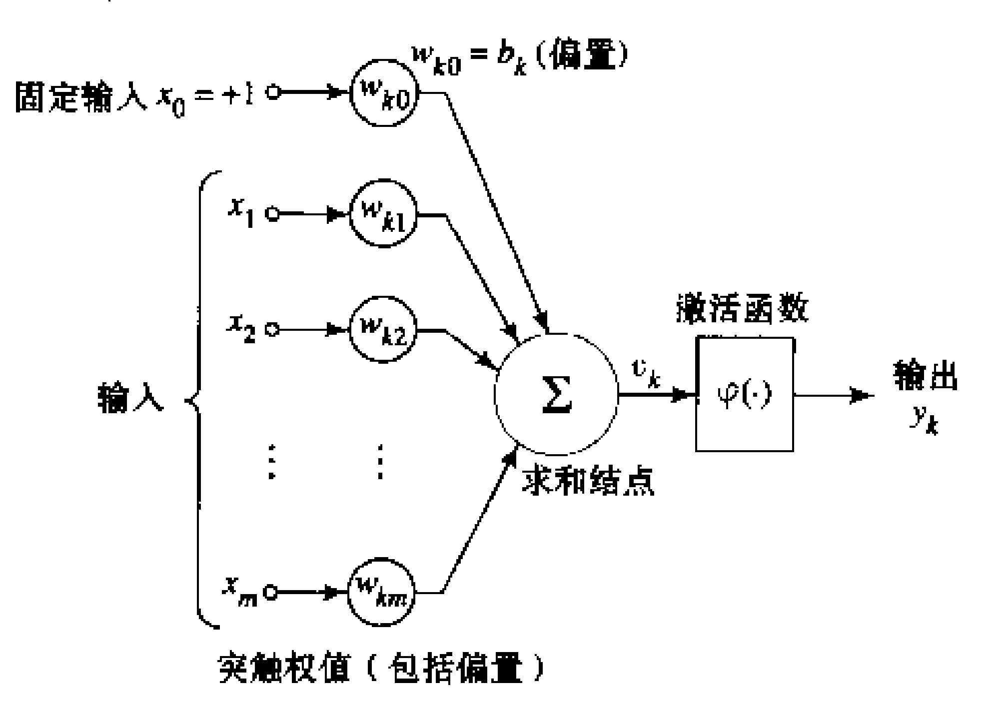


图 1-7 神经元的另一个非线性模型

激活函数的类型

激活函数，记为 $\varphi(v)$，通过诱导局部域 v 定义神经元输出。这里我们给出三种基本的激活函数：

1. 阈值函数。这种激活函数如图 1-8a 所示，可写为：

$$\varphi(v) = \begin{cases} 1 & 如果\ v \geq 0 \\ 0 & 如果\ v < 0 \end{cases} \tag{1.8}$$

在工程文献中，这种函数一般称为 Heaviside 函数。相应地，在神经元 k 使用这种阈值函数，其输出可表示为

$$y_k = \begin{cases} 1 & \text{如果 } v_k \geq 0 \\ 0 & \text{如果 } v_k < 0 \end{cases} \tag{1.9}$$

12

其中 v_k 是神经元的诱导局部域，即

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{1.10}$$

这样一个神经元在文献中称为 McCulloch-Pitts 模型，以纪念 McCulloch and Pitts (1943)的开拓性工作。在模型中，如果神经元的诱导局部域非负，则输出为 1，否则为 0。这描述了 McCulloch-Pitts 模型的皆有或者皆无(all-or-none)的特性。

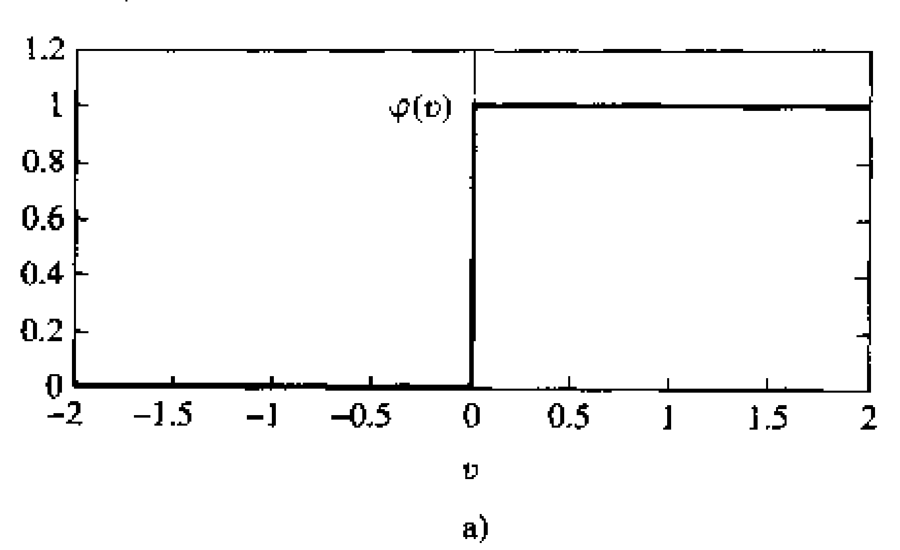
2. 分段线性函数。分段线性函数由图 1-8b 所示，我们有

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & v \leq -\frac{1}{2} \end{cases} \tag{1.11}$$

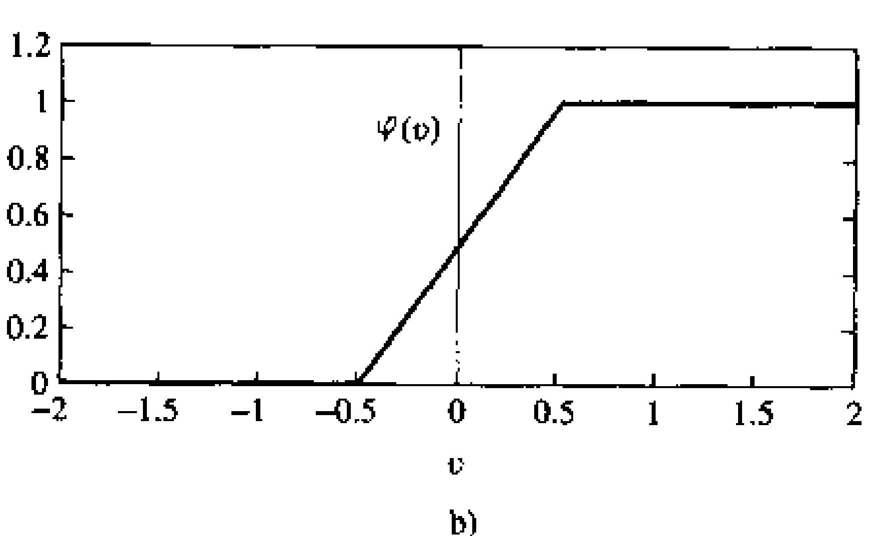
其中，在运算的线性区域内放大因子置为 1。这种形式的激活函数是对非线性放大器的近似。下面两种情况可以看作是此函数的特例：

- 在保持运算的线性区域不超过的情况下，就成为线性组合器。
- 如果线性区的放大因子无穷大，那么此函数退化成阈值函数。

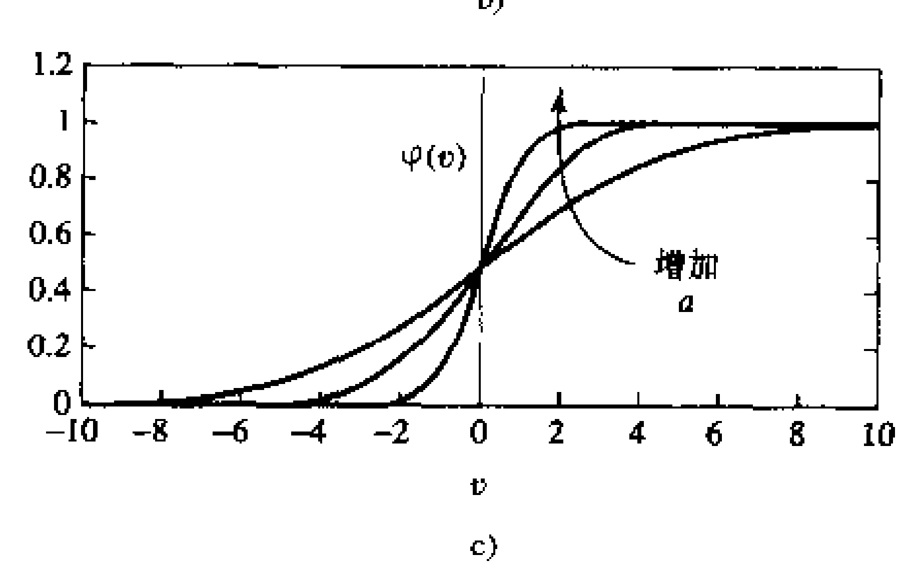
3. sigmoid 函数。此函数的图形是 S-形的，在构造人工神经网络中是最常用的激活函数。它是严格的递增函数，在线性和非线性行为之间显现出较好的平衡[3]。它的一个例子是 logistic 函数[4]，定义如下：



a)



b)



c)

图 1-8
a)阈值函数　b)分段线性函数　c)具有不同倾斜参数 a 的 sigmoid 函数

13

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \tag{1.12}$$

其中 a 是 sigmoid 函数的倾斜参数。改变参数 a 就可以改变倾斜程度，如图 1-8c 所示。实际上，在原点的斜度等于 $a/4$。在极限情况下，倾斜参数趋于无穷，sigmoid 就变成了简单的阈值函数。阈值函数仅取值 0 或 1，而 sigmoid 的值域是 0 到 1 的连续区间。还要注意到 sigmoid 函数是可微分的，而阈值函数不是。(如第 4 章所描述的，可微性是神经网络理论的

一个重要特征。)

在(1.8)，(1.11)和(1.12)中定义的激活函数的值域是0到+1。有时也期望激活函数的值域是-1到+1，这种情况下激活函数是关于原点反对称的；就是说，激活函数是诱导局部域的奇函数。特别地，阈值函数(1.8)的另一种形式是

14

$$\varphi(v)=\begin{cases}1, & \text{如果 } v>0\\ 0, & \text{如果 } v=0\\ -1, & \text{如果 } v<0\end{cases} \tag{1.13}$$

通常称为signum函数。为了和sigmoid函数相对应，我们可以使用双曲正切函数

$$\varphi(v)=\tanh(v) \tag{1.14}$$

如(14)所示，它允许sigmoid型的激活函数取负值，这在分析时是有用的(从第4章可见)。

神经元的统计模型

图1-7的神经元模型是确定性的，它的输入输出行为由所有的输入精确定义。但在一些神经网络应用中，基于随机神经模型的分析更符合需要。用一些解析处理方法，McCulloch-Pitts模型的激活函数用概率分布来实现。特别的，一个神经元允许有两个可能的状态值+1或-1。一个神经元激发(即它的状态开关从"关"到"开")是随机决定的。用x表示神经元的状态，$P(v)$表示激发的概率，其中v是诱导局部域。我们可以设定

$$x=\begin{cases}+1, & \text{以概率 } P(v)\\ -1, & \text{以概率 } 1-P(v)\end{cases}$$

$P(v)$的一个标准选择是sigmoid型的函数(Little，1974)：

$$P(v)=\frac{1}{1+\exp(-v/T)} \tag{1.15}$$

其中T是伪温度，控制激发中的噪声水平即不确定性。但是，不管神经网络是生物的或人工的，它都不是神经网络的物理温度，认识到这一点很重要。进一步，正如所说明的一样，我们仅仅将T看作是一个控制表示突触噪音的效果的热波动的参数。注意当T趋于0，(1.15)所描述的随机神经元就变为无噪声(即确定性)形式，也就是McCulloch-Pitts模型。

1.4 看作有向图的神经网络

图1-5的方框图或图1-7的方框图提供了构成人工神经元模型各个要素的功能描述。我们可以在不牺牲模型功能细节的条件下用信号流图来简化模型外观。Mason(1953,1956)开发了线性网络的一套信号流图，并带有定义好的规则。神经元的非线性限制了它们在神经网络中的应用范围。不过，信号流图在描述神经网络信号流时为我们提供了简洁的方法，我们在本节进行讨论。

信号流图是一个由有向连接(分支)的互连节点组成的网络。一个典型的节点j有一个相应的节点信号x_j。一个典型的有向连接从节点j开始，到k节点结束。它有相应的传递函数或传递系数以确定节点k的信号y_k依赖于节点j的信号x_j之间的方式。图形中各部分的信号流动遵循3条基本规则。

15

规则1 信号仅仅沿着定义好的箭头方向在连接上流动。两种不同的连接可以区别开来：

- 突触连接，它的行为由线性输入输出关系决定。特别如图1-9a所示，节点信号y_k

由节点信号 x_j 乘以突触权值 w_{kj} 产生。

- 激活连接，它的行为一般由非线性输入输出关系决定。如图 1-9b 所示，其中 $\varphi(\cdot)$ 为非线性激活函数。

规则 2 节点信号等于经由进入连接的有关节点的信号的代数和。

这个规则如图 1-9c 所示突触会聚或扇入的情形。

规则 3 节点信号沿每个外向连接向外传递，此时传递的信号完全独立于外向连接的传递函数。

如图 1-9d 所示突触散发或扇出的情形。

16

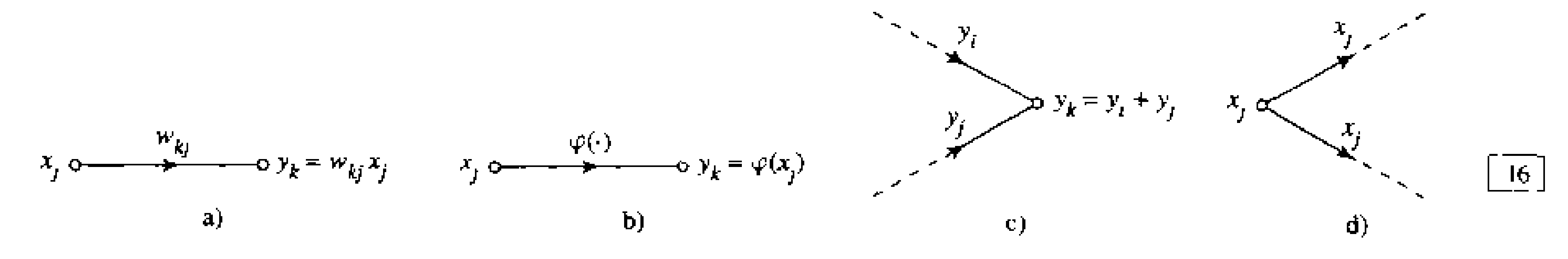


图 1-9 用于构造信号流图的基本规则图示

比如，用这些规则，我们可以制作对应于图1-7的信号流图图 1-10。可以看出，图 1-10 要比图 1-7 的形式更简单，但是它包含了后者描绘的所有功能细节。注意，在两个图中，输入 $x_0 = +1$ 和相关的突触权值 $w_{k0} = b_k$，其中 b_k 是神经元 k 的偏置。

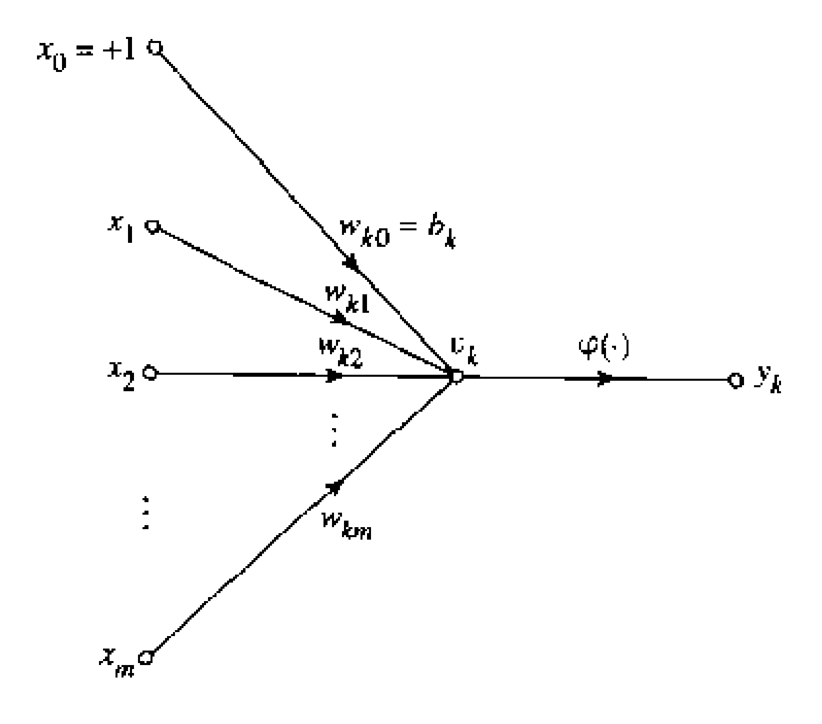


图 1-10 神经元的信号流图

确实，根据图 1-10 的信号流图为神经元模型，我们可以给出一个神经网络的下列数学定义：

神经网络是一个由具有互连接突触的节点和激活连接构成的有向图，具有 4 个主要特征：

1. 每个神经元可表示为一组线性的突触连接，一个应用它的外部偏置，以及可能的非线性激活连接。偏置由和一个固定为 +1 的输入连接的突触连接表示。

2. 神经元的突触连接给它们相应的输入信号加权。

3. 输入信号的加权和构成该神经元的诱导局部域。

4. 激活连接压制神经元的诱导局部域产生输出。

一个神经元的状态可以定义为它的输出信号或者诱导局部域。

一个如此定义的有向图是完全的，这是指它不仅仅描述了神经元间的信号流，也描述了每个神经元内部的信号流。但是当我们的注意集中在神经元之间的信号流上时，可以使用这个图的一个简略形式，它省略神经元内部的信号流的细节。这样的有向图是局部完全的。它的特征是：

1. 源节点向图提供输入信号。

2. 每个神经元由一个计算节点表示。

[17] 3. 联结图中源节点和计算节点之间的通信连接没有权值，它们仅仅提供图中信号流的方向。

这样定义的一个局部完全的有向图就是所谓神经网络的结构图，描述神经网络的布局。图 1-11 给具有 m 个源节点和一个用于偏置的固定为 +1 的节点组成的单一神经元的简单情况。注意表示该神经元的计算节点以阴影显示，而源节点用小方块显示。在本书中，我们都遵循这里的表示方法。在 1.6 节有更精巧的布局结构图的例子。

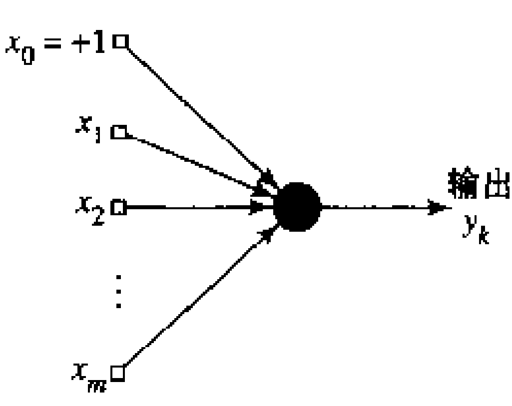


图 1-11 神经元的结构图

总的来说，我们有三种神经网络的图形表示方法：

- 方框图，提供网络的功能描述。
- 信号流图，提供网络中完全的信号流描述。
- 结构图，描述网络布局。

1.5 反馈

反馈存在于动态系统，系统一个元素的输出部分影响作用于该元素输入，因此造成了一个或多个围绕系统的信号传输的封闭路径。实际上，反馈存在于所有动物的神经系统的几乎每一部分中(Freeman，1975)。并且，在一类特殊的神经网络——递归网络的研究中扮演重要的角色。图 1-12 表示单环反馈系统的信号流图，输入信号 $x_j(n)$、内部信号 $x'_j(n)$ 和输出信号 $y_k(n)$ 是离散时间变量 n 的函数。这个系统由"算子"A 表示的前向通路和"算子"B 表示的反馈通路组成，系统是线性的。特别的，前向通道的输出通过反馈通道影响自己的输出。[18] 我们可以很容易得到图 1-12 的输入输出关系：

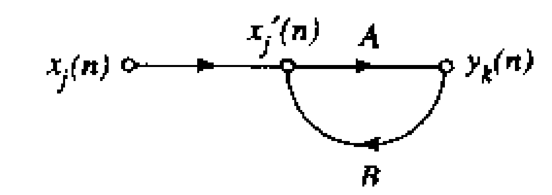


图 1-12 单环反馈系统的信号流图

$$y_k(n) = A[x'_j(n)] \tag{1.16}$$

$$x'_j(n) = x_j(n) + B[y_k(n)] \tag{1.17}$$

其中方括号是为了强调 A 和 B 是扮演算子的角色。在(1.16)，(1.17)中消去 $x'_j(n)$，得到

$$y_k(n) = \frac{A}{1-AB}[x_j(n)] \tag{1.18}$$

我们把 $A/(1-AB)$ 称为系统的闭环算子，AB 称为开环算子。一般说来，开环算子没有交换性，即 $AB \neq BA$。

例如，考虑图 1-13 中的单环反馈系统。A 是一个固定的权值 w；B 是单位延迟算子 z^{-1}，其输出是输入延迟一个时间单位的结果。我们可以将这个系统的闭环算子表示为

$$\frac{A}{1-AB} = \frac{w}{1-wz^{-1}} = w(1-wz^{-1})^{-1}$$

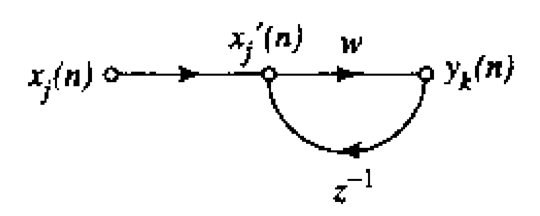


图 1-13 一阶无限冲击响应(IIR)滤波器的信号流图

用 $(1-wz^{-1})^{-1}$ 二项式展开，可以把系统的闭环算子重写为

$$\frac{A}{1-AB} = w\sum_{l=0}^{\infty} w^l z^{-l} \tag{1.19}$$

因此，将式(1.19)代入式(1.18)，我们有

$$y_k(n) = w\sum_{l=0}^{\infty} w^l z^{-l}[x_j(n)] \tag{1.20}$$

其中，再次用方括号强调 z^{-1} 是算子的事实。特别的，由 z^{-1} 的定义我们有

$$z^{-l}[x_j(n)] = x_j(n-l) \tag{1.21}$$

其中 $x_j(n-l)$ 是输入信号延迟 l 个时间单位的样本。因此，可以用输入 $x_j(n)$ 现在的和过去的所有样本的加权和来表示输出 $y_k(n)$：

$$y_k(n) = \sum_{l=0}^{\infty} w^{l+1} x_j(n-l) \tag{1.22}$$

我们现在清楚知道系统的动态行为是由权值 w 控制的。特别是，我们可以识别两种特殊情况：

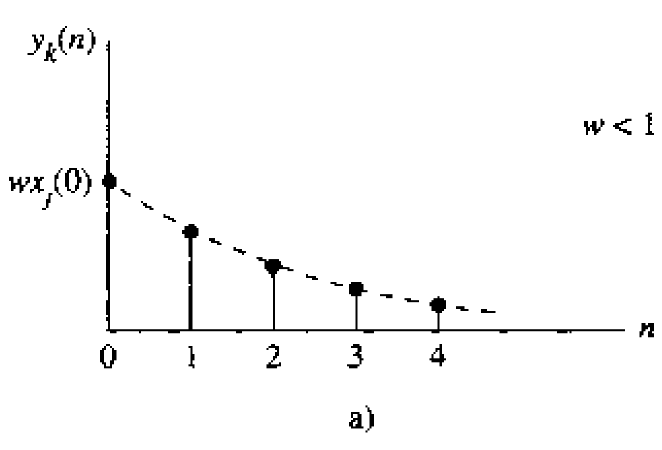
1. $|w| < 1$，此时输出信号 $y_k(n)$ 以指数收敛；也就是说，系统稳定，如图 1-14a 对一个正 w 值的情况所示。

2. $|w| \geqslant 1$，此时输出信号 $y_k(n)$ 发散；也就是说，系统不稳定。图 1-14b 是 $|w| = 1$ 的情况，发散是线性的；图 1-14c 是 $|w| > 1$ 的情况，发散是指数的。

稳定性是反馈系统研究中的突出特征。

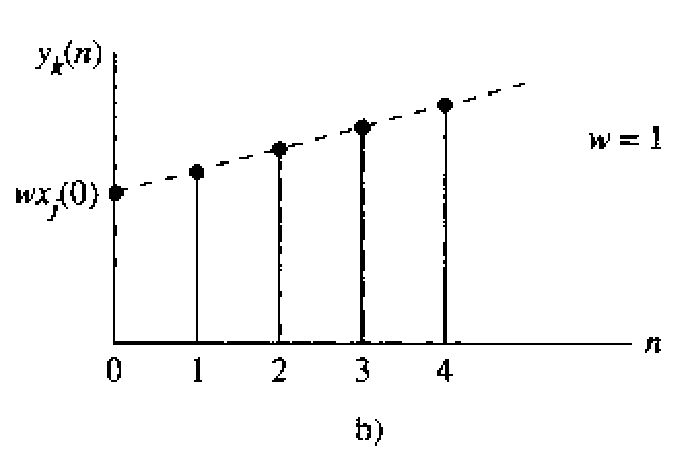
$|w| < 1$ 的情况对应系统具有**无限记忆**，这是指系统的输出依赖于无限过去的输入样本。并且，记忆的强度是随时间 n 指数衰减的。

由于用于构造神经网络的处理单元通常是非线性的，它所涉及的反馈应用的动态行为分析都很复杂。这一点在本书后面部分给出进一步分析。

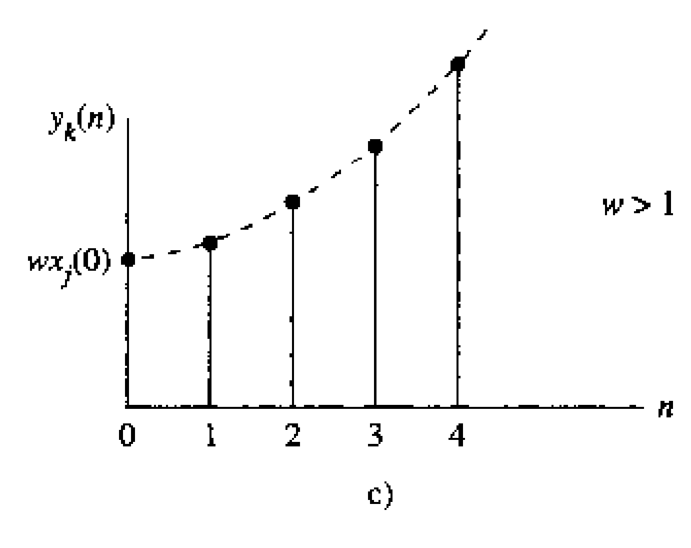


a)

19



b)



c)

图 1-14　图 1-13 中前向权重 w 的三种不同值的时间响应
a)稳定　b)线性发散　c)指数发散

20

1.6　网络结构

神经网络中神经元的构造方式是和训练网络的学习算法紧密连接的。因此，我们可以说，用于网络设计的学习算法(规则)是被构造的。我们将在下一章讨论学习算法的分类，而在本书随后的各章中发展不同的学习算法。这一节我们专注于网络的体系结构。

一般说来，我们可以区分三种基本不同的网络结构。

1. 单层前馈网络

在**分层**网络中，神经元以层的形式组织。在最简单的分层网络中，源节点构成**输入层**，直接投射到神经元**输出层**(计算节点)上去，而不是相反。也就是说，这个网络是严格的**无圈的**或**前馈的**。如图 1-15 所示，输出输入层各有 4 个节点。这样一个网络称为**单层网**。“单层”指的是计算节点(神经元)输出层。我们不把源节点的输入层计算在内，因为在这一层没

有计算。

2. 多层前馈网

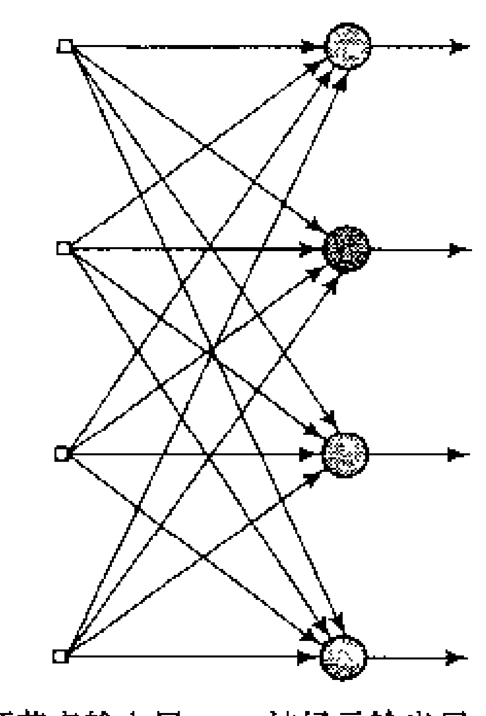


图 1-15 单层前馈或无圈神经元网络

前馈网络的第二种网络有一层或多层隐藏节点层，相应的计算节点称为隐藏单元或隐藏神经元。隐藏神经元的功能是以某种有用方式介入外部输入和网络输出之中。加上一个或多个隐藏层，网络可以引出高阶统计特性。即使网络为局部连接，由于额外的突触连接和额外的神经交互作用，可以使网络在不那么严格意义下获得一个全局关系(Churchland and Sejnowski, 1992)。当输入层很大的时候，隐藏层提取高阶统计特性的能力就更有价值了。

21

输入层的源节点提供激活模式的元素(输入向量)，组成第二层(第一隐藏层)神经元(计算节点)的输入信号。第二层的输出信号作为第三层输入，这样一直传递下去。通常，每一层的输入都是上一层的输出，最后的输出层给出相对于源节点的激活模式的网络输出。结构图如图 1-16 所示。图中只有一个隐藏层以简化神经网络的布局。这是一个 10-4-2 网络，其中有 10 个源节点，4 个隐藏神经元，2 个输出神经元。作为另外一个例子，具有 m 个源节点的前馈网络，第一个隐藏层有 h_1 个神经元，第二个隐藏层有 h_2 个神经元，输出层有 q 个神经元，可以称为 $m-h_1-h_2-q$ 网络。

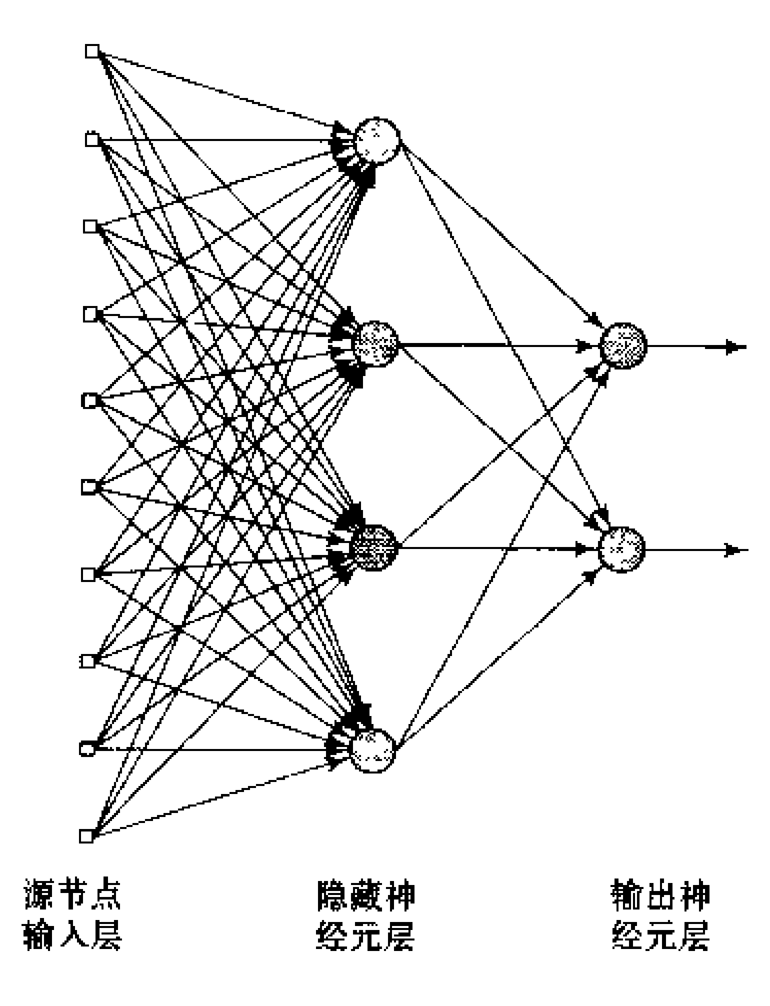


图 1-16 具有一个隐层和输出层的全连接前馈或无圈网络

图 1-16 的网络也可以称之为完全连接网络，这是指相邻层的任意一对节点都有连接。如果不是这样，我们称之为部分连接网络。

22

3. 递归网络

递归网络和前馈网络的区别在于它至少有一个反馈环。例如图1-17所示，递归网络可以是这样，单层网络的每一个神经元的输出都反馈到所有其他神经元的输入中去。这个图中描绘的结构没有自反馈环；自反馈环表示神经元的输出反馈到它自己的输入上去。图 1-17 也没有隐藏层。图 1-18 所示是带有隐藏神经元的一类递归网络，反馈连接的起点包括隐藏层神经元和输出神经元。

反馈环的存在，不管在图 1-17 或图 1-18 的递归结构中，对网络的学习能力和它的性能有深刻的影响。并且，由于反馈环涉及使用单元延迟元素(记为 z^{-1})构成的特殊分支，假如神经网络包含非线性单元，这导致非线性的动态行为。

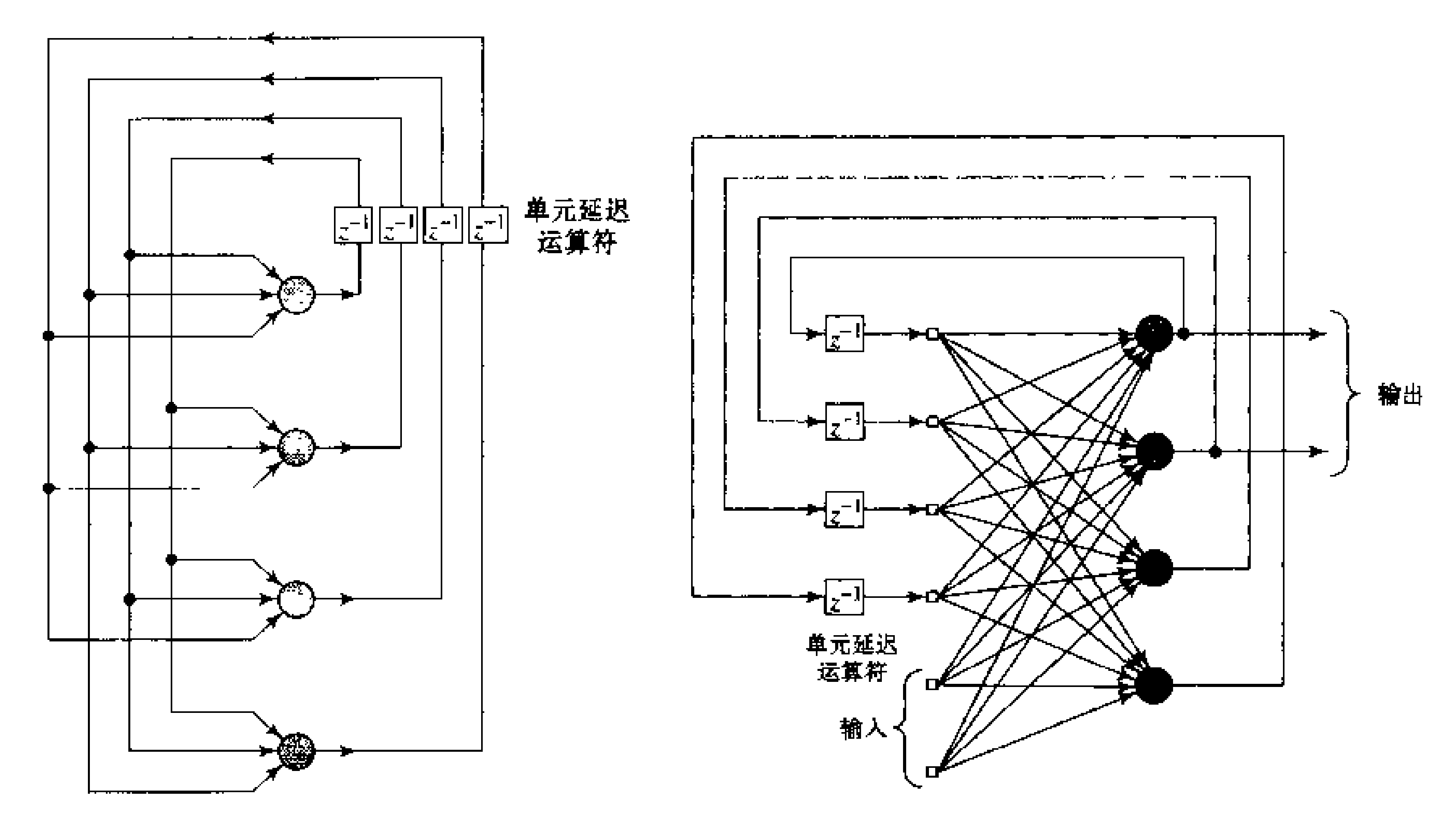


图 1-17　无自反馈环和隐藏神经元的递归网络　　图 1-18　有隐藏神经元的递归网络

1.7　知识表示

在 1.1 节中用到了"知识"这个术语，我们用它来定义神经网络的时候没有对它的涵义作明确的表述。我们注意到这一点，下面给出一般性的定义(Fischler and Firschein,1987)： 23

知识就是人或机器储存起来以备使用的信息或模型，用来对外部世界作出解释、预测和适当的反应。

知识表示的主要特征有两个方面：(1)什么信息是明确表述的；(2)物理上信息是如何被编码和使用的。按知识表示的本性，它是目标导向的。在"智能"机器的现实应用中，可以说好的方案取决于知识的好的表式(Woods,1986)。代表一类特殊智能机器的神经网络也是如此。但是，典型地从输入到内部网络参数的可能表现形式是高度多样性的，这导致基于神经网络的满意解的求解成为一个挑战性的设计。

神经网络的一个主要任务是学习它依存的外部世界(环境)的一个模型，并且保持该模型和真实世界足够相容，这样得到感兴趣的应用的特定目标。有关世界的知识由两类信息组成。

1. 已知世界的状态，由什么事实和已知道什么事实所表示；这种形式的知识被称为先验信息。

2. 对世界的观察(测量)，由设计的探测神经网络所在的运行环境的传感器获得。一般说来，这些观察是带有噪声的，由于传感器的噪声和系统的不完善而产生误差。不管怎样，这样得到的观察是用来训练神经网络例子的信息池。

例子可以是有标记的，也可以是无标记的。例子有标记时，每个例子的输入信号有相应的与之配对的期望响应。另一方面，无标记的例子包括输入信号自身的不同实现。不管怎

[24] 样，一组例子，无论有标号或无标号，代表了神经网络通过训练可以学习的环境知识。

一组由输入信号和相应的期望响应组成的输入输出对称为*训练数据集*或*训练样本*。为了说明怎样使用这样的数据集，例如考虑*手写数字识别问题*。这个问题中，输入信号是一幅黑白图像，每幅图像代表从背景中明显分离的十个数字之一。期望的响应就是"确定"网络的输入信号代表哪个数字。通常训练样本就是手写体数字的大量变形，这代表了真实世界的情形。有了这些样本，可以如下设计网络：

- 第一，选择一个合适的结构，输入层的源节点数和输入图像的像素数一样，而输出层包含10个神经元(每个数字对应一个神经元)。利用合适的算法，以样本的一个子集训练网络。这个设计阶段叫*学习*。
- 第二，用陌生样本检验已训练网络的识别性能。特别，呈现给网络一幅输入图像，此时并不告诉它这幅图像属于哪个数字。网络的性能就用网络报告的数字类别和输入图像的实际的类别的差异来衡量。网络运行的这第二个阶段叫*泛化*，这是借用心理学的术语。

这里神经网络设计与它的传统信息处理对应部分(模式分类器)有着根本的差别。在后一种情况，首先我们通常设计一个环境观察的数学模型，利用真实数据验证这个模型，再以此模型为基础建立设计。相反，神经网络的设计直接基于实际数据，*让数据自己说话*。因此神经网络提供了内嵌于环境的隐含模型，但是也实现了感兴趣的信息处理功能。

用于训练神经网络的例子可以由*正例*和*反例*组成。比如，在被动声纳探测问题上，正例指包括感兴趣的目标(如潜艇)的输入训练数据。在被动声纳环境，我们知道测试数据中海洋生物的可能出现经常造成虚警。为了缓解这个问题，把反例(如海洋生物的回声)包括在训练集中以教会网络不要混淆海洋生物和目标。

在神经网络的独特结构中，周围环境的知识表示由网络的自由参数(即突触权值和偏置)的取值定义。这种知识表示的形式构成神经网络的设计本身，因此，也是网络性能的关键。

[25] 人工网络中的知识表示是很复杂的。但是这里有它通用的4条规则(Anderson,1988)。

规则1 相似的类别中相似输入通常应产生成网络中相似的表示，因此，可以归入同一类中。

度量输入相似性有很多方法。常用的相似度量是利用欧几里德距离。作为特例，令 $\mathbf{x}_i$ 是一个 $m\times 1$ 的实元素列向量，

$$\mathbf{x}_i=[x_{i1},x_{i2},\cdots,x_{im}]^T$$

上标 T 表示矩阵转置。向量 $\mathbf{x}_i$ 就是 m 维空间(称为*欧几里德空间*)的一个点，记为 $\mathbb{R}^m$。两个 $m\times 1$ 向量$\mathbf{x}_i,\mathbf{x}_j$ 之间的*欧几里德距离*就是

$$d(\mathbf{x}_i,\mathbf{x}_j)=\|\mathbf{x}_i-\mathbf{x}_j\|=\left[\sum_{k=1}^{m}(x_{ik}-x_{jk})^2\right]^{1/2} \tag{1.23}$$

其中x_{ik},x_{jk}分别是输入向量$\mathbf{x}_i,\mathbf{x}_j$ 的第 k 个分量。相应地，由向量$\mathbf{x}_i,\mathbf{x}_j$ 表示的两个输入的相似性就定义为欧几里德距离 $d(\mathbf{x}_i,\mathbf{x}_j)$的倒数。输入向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 相距越近，欧几里德距离 $d(\mathbf{x}_i,\mathbf{x}_j)$就越小，相似性就越大。如果两个向量是相似的，规则1说明它们归入同一类。

另一个相似性度量是基于*点积*或*内积*，它借用矩阵代数。给定一对相同维数的向量$\mathbf{x}_i$，$\mathbf{x}_j$，它们的内积就是 $\mathbf{x}_i^T\mathbf{x}_j$，可展开如下：

$$(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j = \sum_{k=1}^{m} x_{ik} x_{jk} \tag{1.24}$$

内积$(\mathbf{x}_i, \mathbf{x}_j)$除以范数积$\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|$，就是两个向量$\mathbf{x}_i, \mathbf{x}_j$的夹角的余弦。

这里定义的两种相似性度量有密切的联系，如图1-19所示。欧几里德距离$\|\mathbf{x}_i - \mathbf{x}_j\|$和向量$\mathbf{x}_i$到向量$\mathbf{x}_j$的“投影”相关。图1-19清楚地表明欧几里德距离$\|\mathbf{x}_i - \mathbf{x}_j\|$越小，向量$\mathbf{x}_i$和$\mathbf{x}_j$越相似，内积$\mathbf{x}_i^T \mathbf{x}_j$越大。

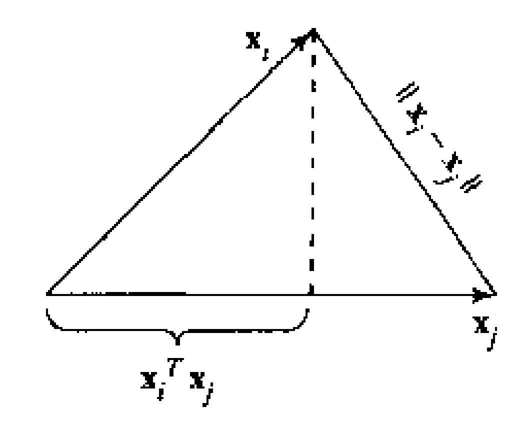


图1-19 图解内积和作为模式相似性度量的欧几里德距离之间的关系

26

为了把这种关系置于形式化基础之上，我们首先将向量$\mathbf{x}_i$和$\mathbf{x}_j$归一化，即

$$\|\mathbf{x}_i\| = \|\mathbf{x}_j\| = 1$$

利用式(1.23)我们就可以写成

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j) = 2 - 2\mathbf{x}_i^T \mathbf{x}_j \tag{1.25}$$

等式(1.25)表明最小化的欧几里德距离$d(\mathbf{x}_i, \mathbf{x}_j)$就对应最大化的内积$(\mathbf{x}_i, \mathbf{x}_j)$和最大化$\mathbf{x}_i$和$\mathbf{x}_j$的相似性。

这里的欧几里德距离和内积的定义都是用确定性的术语定义的。如果向量$\mathbf{x}_i$和$\mathbf{x}_j$是从不同数据总体(池)中得来的，又该怎样定义相似性呢？作为特例，假设两个总体的差异仅在它们的均值向量。令$\boldsymbol{\mu}_i$和$\boldsymbol{\mu}_j$分别表示向量$\mathbf{x}_i$和$\mathbf{x}_j$的均值。也就是说，

$$\boldsymbol{\mu}_i = E[\mathbf{x}_i] \tag{1.26}$$

其中E是统计期望算子。均值向量$\boldsymbol{\mu}_j$同样定义。为了度量这两个总体的距离，我们可以用Mahalanobis距离来衡量，记为d_{ij}。从$\mathbf{x}_i$到$\mathbf{x}_j$的这种距离的平方值定义为(Duda and Hart, 1973)：

$$d_{ij}^2 = (\mathbf{x}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_j) \tag{1.27}$$

其中$\boldsymbol{\Sigma}^{-1}$是协方差矩阵$\boldsymbol{\Sigma}$的逆矩阵。假设两个总体的协方差矩阵是一样的，表示如下：

$$\boldsymbol{\Sigma} = E[(\mathbf{x}_i - \boldsymbol{\mu}_i)(\mathbf{x}_i - \boldsymbol{\mu}_i)^T] = E[(\mathbf{x}_j - \boldsymbol{\mu}_j)(\mathbf{x}_j - \boldsymbol{\mu}_j)^T] \tag{1.28}$$

当$\mathbf{x}_i = \mathbf{x}_j$，$\boldsymbol{\mu}_i = \boldsymbol{\mu}_j = \boldsymbol{\mu}$和$\boldsymbol{\Sigma} = \mathbf{I}$时($\mathbf{I}$为单位矩阵)，Mahalanobis距离变为样本向量$\mathbf{x}_j$和均值向量$\boldsymbol{\mu}$间的欧几里德距离。

规则2 网络对可分离为不同种类的输入向量给出差别很大的表示。

这条规则与规则1正相反。

规则3 如果某个特征很重要，那么网络表示这个向量将涉及大量神经元。

比如，考虑雷达探测涉及在散乱状态(即雷达从不期望的目标如建筑物、树木和云层的反射)下的目标(如航空器)的应用。这样的雷达系统的探测性能由下面两种概率形式来衡量：

27

- *探测概率*，就是目标存在时系统判断目标出现的概率。
- *虚警概率*，就是目标不存在时系统判断目标出现的概率。

按照Neyman-Pearson准则，在虚警概率限制在一定范围的情况下，探测概率达到最大值(Van Trees, 1968)。在这种应用中，收到信号中目标的实际出现代表输入信号中的重要特征。实际上，规则3意味着在真实目标存在的时候应该有大量神经元参与判决该目标出现。按同样道理，仅当散乱状态实际存在的时候才应该有大量神经元参与判决该散乱状态的出现。在两种情形下，大量的神经元保证了判决的高度准确性和对错误神经元的容错性。

规则4 先验信息和不变性应该附加在网络设计中，这样不必学习它们就能简化网络设计。

规则4特别重要，因为真正坚持这一规则就会导致网络具有特殊的(有限制的)结构。这一点是我们正需要的，原因如下(Russo,1991)：

1. 已知生物视觉和听觉网络是非常特别的。

2. 相对于完全连接网络，特殊网络用于调节的自由参数是较少的。因此，特殊网络所需的训练数据更少，学习更快而且常常推广性更强。

3. 通过特殊网络的信息传输速率(即网络的通过数据)是增加的。

4. 和全连接网络相比特殊网络的建设成本比较低，因为规模较小。

怎样在神经网络设计中加入先验信息

当然，怎样在神经网络设计中建立先验信息，以此建立一种特殊的网络结构，这是必须考虑的重要的问题。不幸的是，现在还没有一种有效的规则来使用先验信息提高网络性能；我们只有某些特别的过程，已知可以产生一些有用的结果。特别是，我们使用下面两种技术的结合(LeCun et al.,1990a)：

1. 通过使用称为*接收域*[5]的局部连接，限制网络结构。

2. 通过使用*权值共享*[6]，限制突触权值的选择。

这两种方法，特别是后一种，有很好的附带效益，它使网络自由参数的数量显著下降。

28

作为特例，考虑一个如图1-20所示的部分连接前馈网络。这个网络有带限制的结构。顶部6个源节点组成隐藏神经元1的接收域，网络其余隐藏神经元类推。为满足权值共享限制，我们在隐藏层中每个神经元使用同一组突触权值。这样，对图1-20所示的例子，每个隐藏神经元有6个局部连接，共有4个隐藏神经元，我们可以表示每个隐藏神经元的诱导局部域如下：

$$v_j = \sum_{i=1}^{6} w_i x_{i+j-1}, \quad j = 1,2,3,4 \quad (1.29)$$

其中$\{w_i\}_{i=1}^{6}$构成所有四个隐藏神经元共享的同一权值集，x_k为从源节点$k = i + j - 1$挑选的信号。方程(1.29)为*卷积和*的形式。

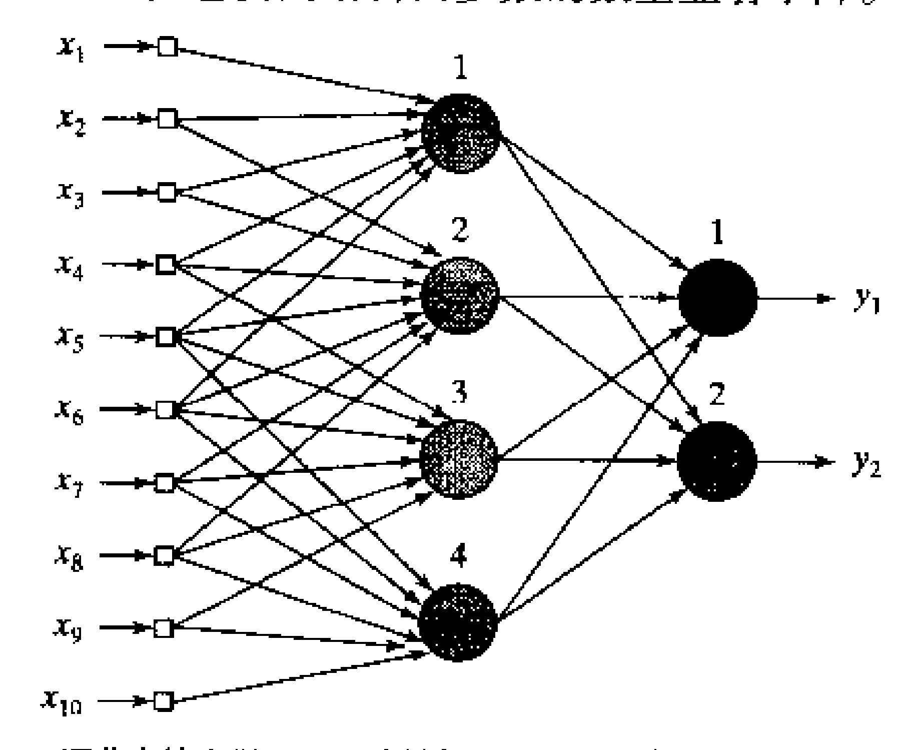


图1-20 联合利用接受域和权值共享的图例。所有四个隐神经元共享它们突触连接的相同权值集

由于这个原因，使用这里描述的局部连接和权值共享的前馈网络称为*卷积网络*。

在神经网络的设计中建立先验信息的问题是属于规则4的一部分；该规则的剩余部分涉及不变性问题。

如何在网络设计中建立不变性

考虑下列物理现象：

- 当感兴趣的目标旋转时，观察者感知到的目标的图像通常会有相应的变化。

- 在一个提供它周围环境的幅度和相位信息的相干雷达中，由于目标相对于雷达射线运动造成的多普勒效应活动目标的回声在频率上会产生偏移。 29
- 人说话的语调会有高低快慢的变化。

为了分别建立一个对象识别系统、一个雷达目标识别系统和一个语音识别系统处理这些现象，系统必须可以应付一定范围内的观察信号的变换。相应地，一个模式识别问题的主要任务就是设计对这种变换不变的分类器。也就是说，分类器输出结果的类别估计不受分类器输入观察信号变换的影响。

至少可用三种技术使得分类器型的神经网络对变换不变(Barnard and Casasent, 1991)：

1. 结构不变性。适当地组织它的设计，在神经网络中加进不变性。特别是，在建立网络的神经元突触连接时要求同一输入变换后必须得到同样的输出。例如考虑利用神经网络对输入图像的分类问题，要求神经网络在平面内不受关于中心的旋转的影响。我们可以在网络中强制加上旋转不变性如下：令 w_{ji} 表示神经元 j 和输入图像的像素 i 的连接。如果对所有两个到图像中心距离相等的像素 i 和 k 强制 $w_{ji}=w_{jk}$，那么神经网络对平面内的旋转不变。但是为了保持旋转不变性，对从原点出发的相同半径距离上的输入图像的每个像素必须复制突触权值 w_{ji}。这指出了结构不变性的一个缺点：神经网络即使在处理中等大小的图像时，网络中的连接数目也会变得非常大。

2. 训练不变性。神经网络有天生的模式分类的能力。利用这种能力可以直接得到下面的变换不变性。用一些来自同一目标的经不同变换后得到的样本(即目标的不同方面)训练网络，这些样本代表着目标的不同变换。假设样本足够大且训练后的网络已经学会分辨目标的不同方面，我们就可以期望训练后的网络能对已出现目标的不同变换作出正确的推广。但是从工程的角度看，训练不变性有两方面不足：第一，如果一个神经网络训练后对已知变换的目标有不变性，不一定能保证它对其他类型的目标的变换也有不变性。第二，网络的计算要求太难对付了，特别在高维特征空间尤其如此。

3. 特征空间不变性。第三种建立神经网络不变性分类器的技术如图 1-21 所示。它依赖于这样的前提条件，即能提取表示输入数据本质信息内容特性的特征，并且它对输入的变换保持不变。如果使用这样的特征，那么分类神经网就可以从刻画具有复杂的判定边界的目标变换范围的负担中解脱出来。确实，同一目标的不同的事例的差异仅仅在于噪音和偶发事件等不可避免因素的影响。特征空间不变性提供了三个明显的好处：第一，适用于网络的特征数可以降低到理想的水平。第二，网络设计的要求放宽了。第三，所有目标的已知变换的不变性都得到保证(Barnard and Casasent, 1991)。但是，这个方法要求所求问题的先验知识。

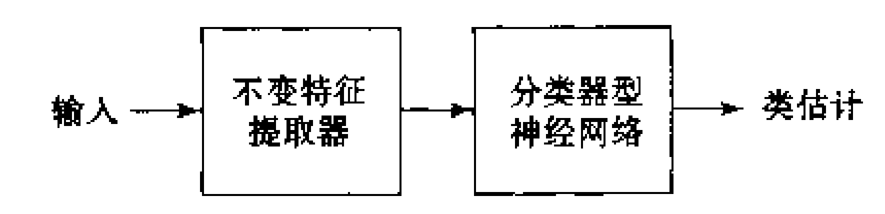


图 1-21　不变性特征空间型系统方框图

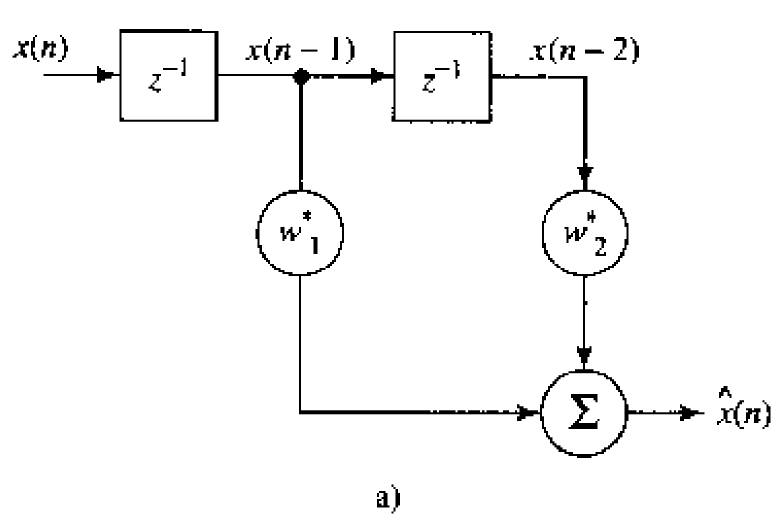
总的说来，利用所描述的不变性特征空间，可以提供最适合神经网络分类器的技术。 30

为了描述不变性特征空间，考虑一个例子，用于空中监控相干雷达系统，其目标可能是飞机，天气，鸟群和地面目标。从这些目标的雷达回声有特有的谱特征。并且，实验研究表明这样的雷达信号容易用阶为中等大小的自回归(autoregressive, AR)过程模型来建模(Haykin and Deng, 1991)。AR 模型是如下对复数数据定义的回归模型的特殊形式：

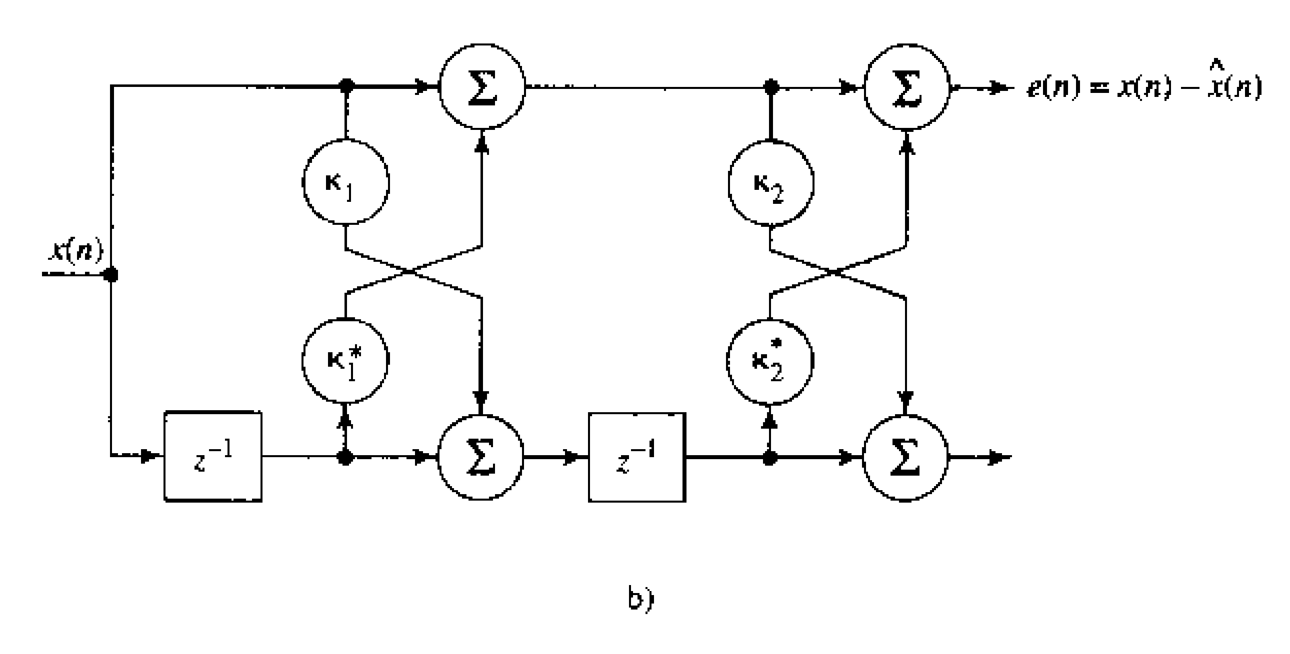
$$x(n) = \sum_{i=1}^{M} a_i^* x(n-i) + e(n) \tag{1.30}$$

其中$\{a_i\}_{i=1}^{M}$为AR系数，M为模型阶，$x(n)$为输入，$e(n)$为白噪声的误差。基本上，方程(1.30)的AR模型由带状延迟线滤波器表示，如图1-22a中$M=2$的情形。同样，它可由图1-22b所示的网格滤波器表示，它的系数称为反射系数。图1-22a中模型的AR系数和图1-22b中模型的反射系数一一对应。所描绘的模型都假设输入$x(n)$是复数，因为在相干雷达的情形，AR系数和反射系数都为复数。在方程(1.30)和图1-22中的星号表示复共轭。现在可以说相干雷达数据可以用一组自回归系数描述，或者由一组相应的反射系数描述。后一组系数有计算上的优点，已存在有效的算法从输入数据直接计算。但是，特征提取问题是很复杂的，因为活动物体产生不同的多普勒频率，这取决测得的物体相对于雷达的径向速度，以及作为特征判别式的反射系数的谱分布会产生模糊。为了克服这种困难，我们必须建立反射系数计算中的多普勒不变性。第一个反射系数的相位角结果与雷达信号的多普勒频率相等。相应地，归一化多普勒频率可以去掉平均多普勒平移的均值。这些可以通过从输入数据计算得到的常规反射系数$\{\kappa_m\}$定义新的反射系数$\{\kappa'_m\}$来实现：

$$\kappa'_m = \kappa_m e^{-jm\theta}, \quad m = 1,2,\cdots,M \tag{1.31}$$



a)



b)

图1-22 二阶自回归模型

a)带状延迟线模型 b)网格滤波器模型(星号表示复共轭)

其中θ为第一反射系数的相位角。(1.31)描述的运算称为外差法。一组多普勒不变雷达特征可由归一化的发射系数$\kappa'_1,\kappa'_2,\cdots,\kappa'_M$表示，$\kappa'_1$为惟一的实系数。我们说过，空中监控的雷达目标主要可归类为飞机、天气、鸟群和地面，前三类目标都是动的，后一种不是。地面回

声混频后的谱参数和飞机的类似。因为其小的多普勒平移，地面回声可以和飞机区别。相应的，雷达分类器包括一个如图1-23所示的后处理器，操作分类结果(编码标号)以识别地面类(Haykin and Deng 1991)。这样，在图1-23中的预处理器处理从分类器输入中抽取的多普勒平移不变特征，而后处理器利用存储的多普勒特征区分返回的地面和飞机信号。 32

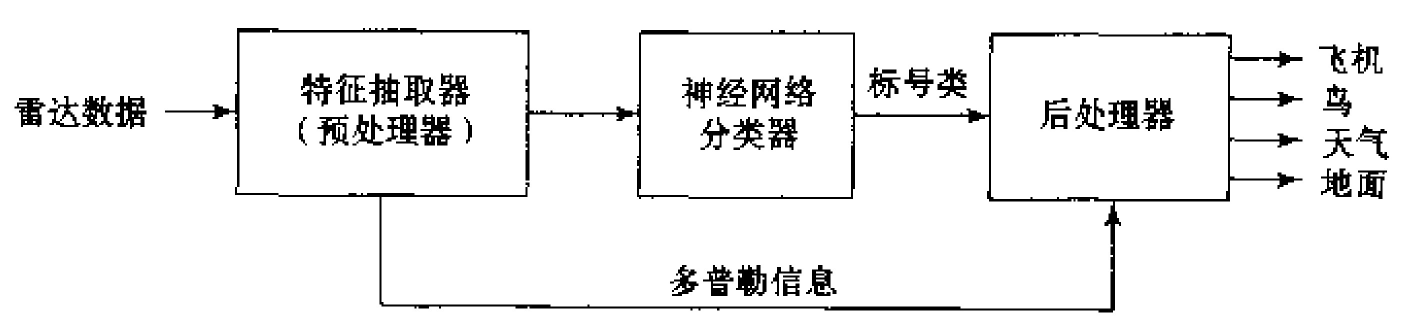


图1-23　雷达信号的多普勒平移不变分类器

神经网络知识表示的一个更有趣的例子是蝙蝠的生物回声定位声纳系统。为了声音映射，大多数蝙蝠使用频率调制(FM或“chirp”)信号，在FM信号中信号的瞬时频率随时间变化。特别的，蝙蝠用口发出短时FM声纳信号，用听觉系统来作接收器。对于感兴趣的目标回声在听觉系统中选用不同声音参数组合的神经元活动来表达。蝙蝠的听觉表达有三个主要的神经维数(Simmons,1991;Simmons and Saillant,1992)：

- *回声频率*，在耳蜗频率图中被编码；通过整个听觉系统的通路保存，按照调制成不同频率的一定神经元的有序排列。
- *回声幅度*，由其他具有不同动态范围的神经元编码；它被表示成幅度调制和每个刺激的放电次数。
- *回声延迟*，通过神经计算编码(基于交叉相关)并产生延迟选择响应。它被表示成目标范围调制。

用于图像形成的目标回声的两个主要特点是目标的“形状”的谱和目标范围的延迟。利用目标不同反射面的回声(反射)的到达时间，蝙蝠感知“形状”。为此目的，回声谱的频率信息被转换为目标的时间结构的估计。由Simmons及其合作者对棕色大蝙蝠(Eptesicus fuscus)进行的试验，严格验证了这个转换过程，它的组成包括并行时域转换和频率对时域转换构成，它的收敛输出产生目标的感知图像的范围轴上的共同延迟。虽然最初执行的回声延迟的听觉时间表示和回声谱的频率表示的方法不同，但看起来蝙蝠的感知协调性归因于变换自身的一些性质。并且特征不变性被嵌入声纳图像形成过程，所以它本质上独立于目标相对运动和蝙蝠自己的运动。

回到本节主题上来，即神经网络中的知识表示，这个论题和1.6节描述的网络结构有直接关系。不幸的是，还没有成功的理论可以根据环境优化神经网络结构，或者评价修改网络结构对网络内部知识表示的影响。实际上，对这些问题的满意结果经常要用穷尽试验研究来得到，这样神经网络的设计者也是结构学习环中的关键部分。 33

不管如何完成设计，对于感兴趣的问题领域的知识，总是以相当简单和直接的方式通过对网络的训练来得到的。这样获得的知识，网络通过突触连接的权值以简洁的分布式形式表示。这种形式的知识表示使得神经网络可以改进和推广，不幸的是神经网络受到它固有的缺乏解释能力的困扰，即不能以综合的方式解释作出决定或报告输出结果的计算过程。这是一个严重的局限，特别是对于那些主要关注安全的任务，比如空中交通管制和医疗诊断。在这

类应用中，提供某种形式的解释能力不仅是非常期望的，而且是绝对需要的。提供这种功能的一个方法是把神经网络和人工智能集成一个混合系统，这在下节讨论。

1.8 人工智能和神经网络

人工智能(AI)的目的是给完成人类当前更胜任的感知任务的机器提供范例或算法。这就是Sage在1990年采用的关于AI的陈述。请注意，这并不是人工智能惟一公认的定义。

一个AI系统必须可以完成三种工作：(1)储备知识，(2)使用储备知识解决问题，(3)通过经验获得新知识。一个AI系统有三个关键部分：表示，推理和学习，如图1-24所示。

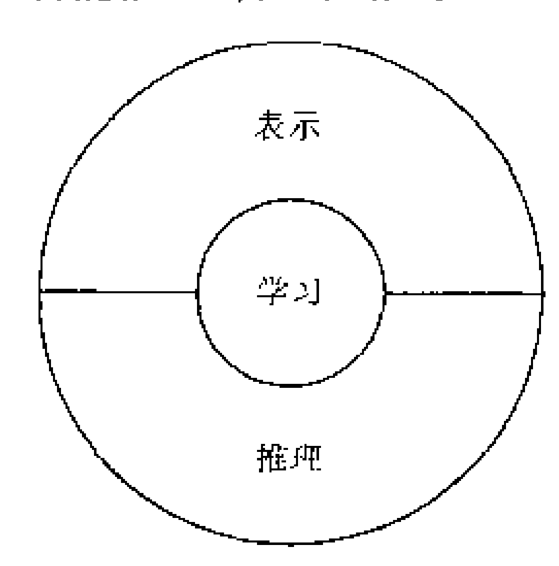


图1-24 AI系统的三个关键组成部分

1. 表示。也许AI最显著的特征就是大量使用符号结构语言表达感兴趣的问题领域的一般知识和问题求解的特殊知识。这些符号通常以常见的形式用于公式中，使得使用者比较容易理解AI的符号表式。确实，AI明确的符号使得它很适合人机交流。

34

AI研究人员所使用的"知识"只不过是数据的另一种名称，它可以是说明性的，也可以是程序的。在说明表示中，知识用一种静态的事实集合以及相应的一小组操作这些事实的通用程序构成。说明表示的一个代表特征是在使用者眼中它自身拥有意义，而与它们在AI系统用途无关。另一方面，在程序表示中，知识嵌入一种可执行代码中，由代码表达知识的含义。这两种形式的知识，不管是说明性的或程序的，在大多数问题领域中都是需要的。

2. 推理。在它最基本的形式中，推理是解决问题的能力。一个可以称为推理系统的系统必须具备一定条件(Fischler and Firschein, 1987)：

- 系统必须能够表示和解决广泛领域内的问题和问题类型。
- 系统必须能够利用它所知道的明确的或隐含的信息。
- 系统必须有一个控制机制，可以决定解决特定问题时使用哪些操作，什么时候已经获得问题的一个特定解，或者什么时候应该中止问题的进一步工作。

求解中的问题可被看作一个搜索问题。处理"搜索"的通用方法是使用规则、数据、控制(Nilsson, 1980)。规则作用于数据，而控制作用于规则。考虑一个例子，"旅行商问题"要求是找出最短的周游各个城市且每个城市仅经过一次的旅行线路。这个问题的数据由可能的线路集和费用的加权图构成，规则决定从一个城市到另一个城市的路径，控制决定在何时使用什么规则。

在现实中遇到的很多情况(如医疗诊断)，可用知识是不完整和不准确的。这时使用概率推理程序，从而允许AI系统可以处理不确定性(Russell and Norvig, 1995; Pearl, 1998)。

3. 学习。在图1-25所示的简单机器学习模型中，环境向学习单元提供信息，学习单元用这些信息来改进知识库，最后性能单元使用知识库完成它的任务。环境给予机器的这些信息通常是不完善的，所以学习单元不能事先知道如何补充遗漏的细节或忽略不重要的细节。因而系统只能凭猜测开始运行，然后接收性能单元的反馈。反馈机制可以使机器评价它的假设并作出必要的修正。

35

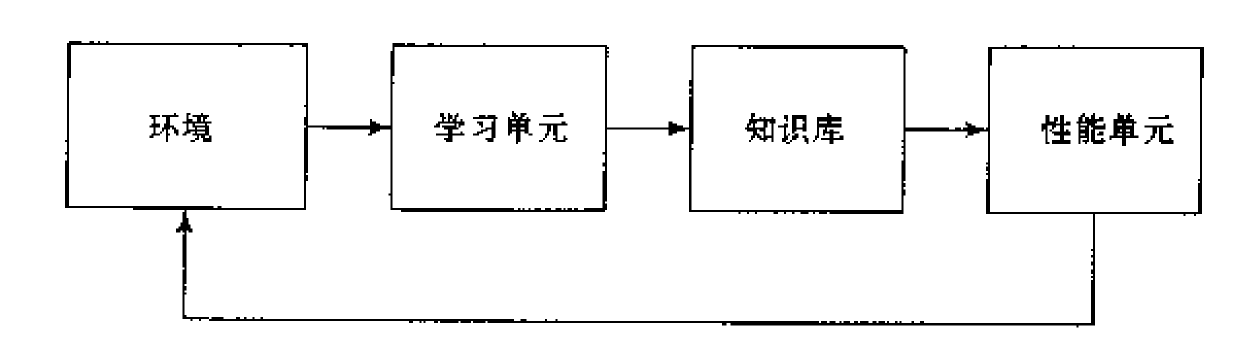


图 1-25 机器学习的简单模式

机器学习涉及两种很不一样的信息处理：归纳和演绎。归纳信息处理，一般模式和规则由原始数据和经验决定。在演绎信息处理中，一般的规则被用来得出特定的事实。基于相似性的学习使用归纳，但是定理的证明是公理和已知定理的演绎。基于解释的学习同时使用演绎和归纳。

知识库的重要性和学习中的困难使得发展各种方法增加知识库。特别是，如果在给定领域有专家，那么取得编辑好的专家经验比试图复制和亲自经历获得经验的过程要容易得多。实际上，这就是专家系统的思想。

怎样将熟悉符号的 AI 机器与作为认知模型的神经系统进行比较？为了这个比较，我们作下面 3 分支：解释水平，处理风格和表示结构(Memmi，1989)。

1. *解释水平*。传统的 AI 中，重点是建立*符号表示*，这样称呼大概是因为它们代表某些事物。从认知的观点，AI 假设存在心理表示，并且它以符号表示的*顺序处理*对认知建模(Newell and Simon，1972)。

另一方面，神经网络强调的重点是*并行分布式处理*(PDP)模型的发展。这些模型假定信息处理通过大量神经元间的相互作用来进行，网络中每个神经元发送兴奋或抑制信号给其他神经元(Rumelhart and McClelland，1986)。同时，神经网络更强调认知现象的神经生物学解释。

2. *处理风格*。在传统的 AI 中，如同在典型的计算机程序中一样处理是*串行的*。即使在没有事先确定的顺序(例如扫描专家系统的事实和规则)的情况下，处理还是一步一步进行的。串行处理的灵感最可能来自自然语言和逻辑推理的串行性以及 von Neumann 机器的结构。不应忘记，传统的 AI 在 von Neumann 机器之后不久诞生的，它和 von Neumann 机器有着相同的智力纪元。

相反，*并行性*在概念上不仅是神经网络信息处理的本质，也是它们灵活性的来源。并且并行性是大规模的(几十万个神经元)，这给予神经网络一个很好的鲁棒性。计算被扩展到许多神经元网络中，个别神经元的状态同它们的期望值偏离并不重要。噪音输入或者不完全的输入也可以被识别，受损网络也可以满意工作，并且学习不必完美。网络的性能在一定范围内缓慢下降。网络甚至可以通过“粗编码”而更加健壮(Hinton，1981)，这里每个特征散布在几个神经元上。

36

3. *表示结构*。传统的 AI 追求思维的语言为模型，我们发现符号表示具有*拟语言结构*。像自然语言的表示一样，经典的 AI 表示一般很复杂，它由简单符号以系统化方式建立。给定有限的符号集，有意义的新表达式可能由符号表达式的组合性以及语法结构和语义的类比构成。

表示的本质和结构是神经网络的关键问题。在 1988 年 3 月《*Cognition*》(识知)杂志的特刊上，Fodor 和 Pylyshyn 有力地批评了神经网络在处理认知和语言中的计算适宜性。他们表示

神经网络处在两个基本的认知问题的错误一边上：心理表示的性质和心理过程的本质。按他们的观点，对于传统的AI而言：

- 心理表示独特地表达了组合的组成结构和组合语义。
- 心理过程对它们操作的表示的组合结构特别敏感。

但是，这不是对神经网络来说的。

总起来说，我们可以把符号化AI描述成是算法语言和数据表示的由顶向下的形式操作。但是，可以把神经网络描述成是具有天生的学习能力的并行分布式处理器，通常以由底向上的方式运行。对于实现认知任务，看起来不能单独使用基于符号AI或神经网络寻求答案，一个更有效的方法是集成二者，建立结构化的连接论者模型或混合系统。这样做，我们能组合神经网络的期望自适应性、鲁棒性及一致性特征以及符号AI固有的表示、推理及通用性特征(Feldman,1992;Waltz,1997)。实际上，基于这个目标，已开发出从训练过的神经网络中抽取规则的方法。除了理解怎样集成符号和连接论者方法以建立智能机器，从神经网络中抽取规则还其他几个原因(Andrews and Diederich,1996)：

- 用户接近和理解神经网络的内部状态有助于确认软件系统中神经网络组件的正确性。
- 通过(1)辨别没有适当表示的训练数据在输入空间中区域，或(2)指明神经网络可能无法推广的环境，提高神经网络的泛化能力。
- 发现用于数据探索(挖掘)的输入数据的潜在特征。
- 在智能机器开发中提供穿越连接论者方法和符号方法的边界的手段。
- 在安全性为必要条件的特殊类型的系统中满足安全的严格需要。

37

1.9 历史注释

我们用一些历史注释[7]结束这一章对神经网络的介绍。

现代的神经网络开始于McCulloch and Pitts(1943)的开拓性工作。McCulloch被培养成精神病学家和解剖学家。他用20年的时间考虑神经系统中关于事件的表示问题。Pitts是数学天才，于1942年加入McCulloch的工作。根据Rall(1990)，McCulloch和Pitts 1943年写的论文在一个神经建模小组公布时，该小组在Rashevsky领导下在芝加哥大学至少五年前就很活跃了。

在他们的经典论文里，McCulloch和Pitts结合了神经生理学和数理逻辑的研究描述了一个神经网络的逻辑演算。他们的神经元模型假定遵循一种所谓“有或无”(*all-or-none*)规则。如果如此简单的神经元数目足够多和适当设置突触连接并且同步操作，McCulloch和Pitts证明这样构成的网络原则上可以计算任何可计算函数。这是一个有重大意义的结果，有了它就标志着神经网络和人工智能学科的诞生。

McCulloch和Pitts 1943年的论文从那时直到现在一直被广泛阅读。它影响了von Neumann，使得他在EDVAC(Electronic Discrete Variable Automatic Computer，电子离散变元自动计算机)中，使用从McCulloch和Pitts的神经元导出的理想化开关延迟元件，这台机器是从ENIAC(Electronic Numerical Integrator and Computer，电子数值积分器和计算机)发展而来的(Aspray and Burks,1986)。ENIAC是第一台通用电子计算机，从1943年到1946年在宾夕法尼亚大学摩尔电子工程学院建成。McCulloch-Pitts的形式化神经网络理论，在von Neumann 1949年在Illinois大学所作的四个报告的第二个报告中成为主要内容。

1948年，Wiener的名著《*Cybernetics*》(控制论)出版，为控制、通信和统计信号处理描述

了某些重要概念。1961年该书第二版出版发行，添加了关于学习和自组织的新材料。在第二版的第2章中，Wiener看来在主题方面抓住了统计力学的物理意义，但是把统计力学和学习系统连系起来获得丰硕成果，却留给了Hopfield(在30多年以后)。

神经网络第二个重要发展是在1949年Hebb的书《*The Organization of Behavior*》(行为组织学)出版，他在书中第一次清楚说明了突触修正的生理学学习规则。特别是，Hebb提出人脑的连接方式在机体学习不同功能任务时是连续变化的，神经组织就是通过这种变化创建起来的。Hebb继承了Ramón y Cajál早期的假设并引入自己现在著名的学习假说，即两个神经元之间的可变突触的作用被突触两端神经元中一个对另一个的重复的激活加强了。Hebb的书在心理学家中有巨大的影响，但遗憾的是对工程界影响很少或没有影响。

Hebb的书是学习系统和自适应系统的计算模型发展的灵感源泉。Rochester, Holland, Haibt and Duda的论文(1956)，也许是用计算机模拟测试以Hebb学习假说为基础的严格公式化的神经理论的第一次尝试；论文报告的模拟结果表明必须加上抑制理论才能实际工作。同一年，Uttley(1956)演示了带有可修改的突触的神经网络，可以学习分类简单的二值模式集。Uttley引入了所谓泄漏集成和点火神经元(leaky integrate and fire neuron)，后来Caianiello(1961)对它进行了形式化分析。在再较晚的工作中，Uttley(1979)假设了神经系统可变突触的作用依赖于突触两端波动状态的统计关系，因此和Shannon的信息论联系起来。 [38]

1952年Ashby的书《*Design for a Brain: The Origin of Adaptive Behavior*》(脑的设计：自适应行为的起源)出版，今天读起来和过去一样也是引人入胜的。这本书关注的是基本概念，即自适应行为不是于与生俱来而是后天学习的，通过学习动物(系统)的行为变得更好。这本书强调活的机体如同机器的动态方面和有关稳定性的概念。

1954年Minsky在普林斯顿大学写了"神经网络"的博士论文，题目是"Theory of Neural-Analog Reinforcement Systems and Its Application to the Brain-Model Problem"。1961年Minsky发表了早期关于AI的优秀论文"Steps Toward Artificial Intelligence"，后面这篇文章包括了有关现在称为神经网络内容的一大节。1967年Minsky出版了《*Computation: Finite and Infinite Machines*》(计算：有限和无限机器)这本书。它是第一本以书的形式扩展了McCulloch和Pitts 1943年的结果，并把它们置于自动机理论和计算理论的背景中。

也是在1954年，Gabor提出了非线性自适应滤波器的思想，他是早期通信理论的先驱者之一和全息照相术的发明者。他接着在合作者的帮助下致力于建立这样的机器，其细节描述在Gabor et al.(1960)中。通过把随机过程样本以及希望机器产生的目标函数一起提供给机器来完成学习。

20世纪50年代，Taylor(1956)开始研究联想记忆。接着Steinbuch(1961)引入了学习矩阵；这个矩阵由插在成行的"感觉"接收器和"马达"效应器之间的开关平面网络构成。在1969年，Willshaw, Buneman和Longuet-Higgins发表了关于非全息照相术的联想记忆的优秀论文。这篇文章给出了两类网络模型：实现相关矩阵的简单光学系统和由光学记忆提出的与之相关的神经网络。联想记忆早期发展的其他重要贡献包括Anderson(1972)，Kohonen(1972)和Nakano(1972)的文章，他们在同一年在外积学习规则的基础上独立地引入相关矩阵记忆的思想。

Von Neumann是20世纪前50年的科学巨匠。数字计算机设计的基础von Neumann结构为了纪念他而命名的。1955年耶鲁大学邀请他在1956年作Silliman报告。他死于1957年，稍后他的未完成的Silliman报告手稿出版成书：《*The Computer and the Brain*》(计算机和人脑， [39]

1958)。这本书很有意思，因为它提示了如果 von Neumann 不死他会做什么；他开始意识到人脑和计算机的巨大差异。

神经网络中特别关心的一个问题是利用被认为不可靠的神经元部件构建可靠的神经网络。von Neumann(1956)利用冗余的思想解决了这个重要的问题，这种思想使得 Winograd 和 Cowan(1963)建议在神经网络中使用*分布式冗余表示*。他们证明大量的元件怎样能集体表示增加鲁棒性和并行性的单个概念。

在 McCulloch 和 Pitts 的经典论文发表 15 年以后，Rosenblatt(1958)在他有关*感知器*的研究中提出了模式识别问题的新方法，一种新的有监督学习方法。所谓的*感知器收敛定理*使 Rosenblatt 取得巨大的成功。Rosenblatt(1960b)年概述了感知器收敛定理的第一个证明；该定理的证明也出现在 Novikoff(1963)和其他人的工作中。Widrow 和 Hoff 引进了*最小均方*(LMS)算法并用它构成了 Adaline(adaptive linear element，*自适应线性元件*)。感知器和 Adaline 的区别在于训练过程。最早的可训练的具有多个自适应元件的分层神经网络之一是由 Widrow 和他的学生提出的 Madaline(multiple-adaline)结构(Widrow,1962)。1967 年 Amari 把随机梯度方法用于模式分类。1965 年 Nilsson 出版《*Learning Machines*》(学习机器)一书，迄今为止仍是一本用超平面区分线性可分模式的最好的著作。在 20 世纪 60 年代感知器的经典时期，好像神经网络可以做任何事。但是，随之而来的 Minsky 和 Papert(1969)的书，利用数学证明单层感知器所能计算的根本局限。在有关多层感知器的简短一节中，他们认为没有任何理由假定单层感知器的任何局限可以在多层的情况下被克服。

在多层感知器的设计中面临一个重要的问题就是*信任赋值问题*(即隐藏神经元在网络中的信任赋值问题)。Minsky(1961)在他的“增强学习系统的信任赋值问题”中首次使用了“信任赋值”术语。在 60 年代末，提出了解决感知器的信任赋值问题所必需的大多数的思想和基本概念，如像现在称之为 Hopfield 网络的递归(吸引子神经)网络所固有的许多基本思想。然而，直到 80 年代这些基本问题的解才出现。根据 Cowan(1990)，十多年的这种推延主要有三个原因：

40

- 一个原因是技术性的——没有个人电脑或工作站作实验。例如，当时 Gabor 发明了他的非线性学习滤波器，而他的研究组为此花了另外六年多的时间用模拟装置建立了一个滤波器(Gabor,1954;Gabor et al.,1960)。
- 另外的原因部分是心理上的，部分是经费上的。Minsky 和 Papert 在 1969 年的专题论文当然不鼓励任何人开展感知器的研究工作或一些机构去支持他们研究。
- 在神经网络和栅格自旋之间的类比还未成熟。直到 1975 年 Sherrington 和 Kirkpatrick 才发明了*自旋玻璃网模型*。

在 70 年代这些因素以这种或那种方式阻碍了人们进一步研究神经网络。除了一些心理学和神经科学方面的专家之外，许多研究人员在那个时期都改变了研究领域。确实只有屈指可数的早期开创者继续神经网络研究。从工程学的角度，我们可以回过头来将 70 年代视为神经网络的潜伏期。

在 70 年代出现的一个重要活动就是利用竞争学习的*自组织映射*。von der Malsburg(1973)完成的计算机模拟工作也许是第一次演示了自组织。在人脑中拓扑有序映射启发下，1976 年 Willshaw 和 von der Malsburg 发表了第一篇关于自组织映射形成的论文。

在 80 年代神经网络的理论和设计主要是在几个前沿方面取得了成绩，随之神经网络的

研究工作进入了恢复期。

Grossberg(1980)，基于他的竞争学习理论的早期工作(Crossberg,1972,1976a,b)，建立了一个新的自组织原则，就是著名的*自适应共振理论*(adaptive resonance theory,ART)。基本上说，这个理论包括一个由底向上的识别层和一个由顶向下的产生层。如果输入模式和已学习的反馈模式匹配，一个叫做"自适应共振"的动态状态(即神经活动的放大和延长)就会发生。这个*前向/反向映射原则*已由其他的研究者在不同的条件下重新发现。

在1982年，Hopfield用能量函数的思想形成一种了解具有对称突触连接的递归网络所执行的计算的新方法。并且他在这种递归网络和统计物理中使用的Ising模型之间建立了同构。这个类比为一连串的物理理论(和物理学家)进入到神经元模型铺平了道路，因此神经网络的领域变化了。这类具有反馈的特殊神经网络在80年代引起了大量的关注，在那个时期产生了著名的Hopfield网络。尽管Hopfield网络可能不是真正的神经生物系统模型，它们包涵的原理(即在动态的稳定网络中存储信息的原理)是极深刻的。事实上，这个原理可以追溯到许多其他研究者的开拓性工作：

- Cragg and Tamperley(1954,1955)从观察得出，正是由于神经元能被"点火"(激活)或"不点火"(静止)，所以在一个栅网中的原子可以使它们自旋指向"上"或"下"。
- Cowan(1967)引入了"sigmoid"激活特征和一个神经元基于logistic函数的平滑激活条件。
- Grossberg(1967,1968)引入了一个神经元的*加性模型*，涉及非线性差分/微分方程，并且探索了作为短期记忆为基础的模型用途。 [41]
- Amari(1972)独立地引入了神经元的加性模型，并用它研究随机连接的类神经元的元件的动态行为。
- Wilson,Cowan(1972)推导了包括兴奋和抑制模型神经元的空间局部化的群体动力学耦合非线性微分方程。
- Little and Shaw(1975)描述了神经元激活或不激活的*概率模型*，并用它发展了短期记忆理论。
- Anderson Silverstein,Ritz and Jones(1977)提出*盒中脑状态*(brain-state-in-a-box,BSB)模型，由一个耦合非线形动力学的简单联想网络组成。

因此毫不奇怪，1982年Hopfield的论文发表后引起了很大争论。不过，该论文第一次使在动态的稳定网络中存储信息的原理清楚了。Hopfiled表明了他对从统计力学自旋玻璃模型检验具有对称连接的特殊递归网络富有洞察力，对称性设计可以保证收敛到一个稳定的条件。1983年，Cohen和Grossberg建立了包括时间连续Hopfield网络作为特例的评价*按内容寻址记忆*的一般原则。吸引子神经网络的一个与众不同的特征，是以自然的方式证明自己处于网络的非线性动力学中，用这种方式，时间是学习的重要维数。在这个背景下Cohen-Grossberg的定理非常重要。

1982年另一个重大发展是Kohonen关于使用一维或二维格网结构的自组织映射研究的文章，这在某些方面与Willshaw和von der Malsburg稍早的工作不同。在文献中Kohonen工作在分析和应用方面比Willshaw和von der Malsburg的模型得到了更多的注意，已经成为这一领域其他创新的评估标准。

1983年Kirkpatrick，Gelatt和Vecchi描述了解决组合最优化的问题的称为*模拟退火*的新方法。模拟退火植根统计力学，是基于Metropolis et al.(1953)在计算机仿真中首先使用的一

种简单技术。Ackley，Hinton and Sejnowski(1985)利用模拟退火的思想发展称为Boltzmann机的随机机器，它是多层神经网络的第一个成功实现。虽然证明Boltzmann机的学习算法没有反向传播算法的计算效率高，但它证明了Minsky and Papert(1969)的猜想是不成立的，打破了心理障碍。Boltzmann机也为Neal(1992)随后的sigmoid信度网络的发展作了铺垫工作。sigmoid 42 信度网络完成了两件事：(1)学习显著改善；(2)联系了神经网络和信度网络。sigmoid信度网络学习性能的进一步提高是Saul，Jakkolla and Jordan(1996)利用一个植根于统计力学的平均场理论作出的。

Barto，Sutton和Anderson一篇关于增强式学习的论文发表于1983年。虽然他们不是第一次使用增强式学习(例如Minsky在他1954年的博士论文中考虑过它)，但这篇文章引起了关于增强式学习及其在控制中应用的极大兴趣。特别是，他们证明了一个增强式学习系统可以在没有帮助教师的情况下学习平衡倒立摆(broomstick，即车上立的杆)。学习系统仅要求当杆对竖直方向倾斜超过一定角度或车到达轨道的端点时发出失败信号。1996年Bertsekas和Tsitsiklis的著作《*Neuro-dynamic*》(神经-动态规划)出版。这本书把增强式和Bellman的动态规划相联系，把它放在一个恰当的数学基础上。

1984年Braitenberg的书《*Vehicles: Experiments in Synthetic Psychology*》(工具：综合心理学的实验)出版。在这本书中Braitenberg提出了目标导向的自组织行为原则：利用公认的基本机制的综合而非由顶向下的分析是最好了解一个复杂过程的方法。在科幻小说的形式下，Braitenberg通过描述各种具有简单内部结构的机器说明了这个重要原则。他对这样一个主题直接或间接研究了二十多年：这些机器的特性和它们的行为受到有关动物脑的事实的启迪。

1986年Rumelhart，Hinton和Williams报告了反向传播算法的发展。同一年，由Rumelhart和McClelland编辑的著名的两卷集著作《*Parallel Distributed Processing: Explorations in the Microstructures of Cognition*》(并行分布式处理：认知微结构的探索)出版。后一本书在反向传播算法的使用方面产生重大影响，它已成为最通用的多层感知器的训练算法。事实上，反向传播学习在同一时间在其他两个地方被独立发现(Parker，1985；LeCun，1985)。在80年代中期发现反向传播算法后，获悉Harvard大学的Werbos早在1974年8月的博士学位论文已经描述了；Werbos的博士论文是描述有效的反转模式梯度计算的第一篇文献，它被用于以神经网络作为特例的一般网络模型。反向传播的基本思想可进一步追溯到Bryson和Ho(1969)的书《*Applied Optimal Control*》(应用最优控制)。在该书标题为"多阶段系统"的2.2节中，描述了使用Lagrange形式的反向传播推导。但是，最终的分析得出反向传播算法的许多荣誉属于Rumelhart，Hinton和Williams(1986)，因为他们提出了它在机器学习中的应用并且演示了它怎样工作。

1988年Linsker描述了认知网络中自组织问题的新原理(Linsker，1988a)。这个原理被设计 43 成保持有关输入活动模式的最大信息，以这样的约束限制突触连接和突触动态范围。其他几位视觉研究者也提出了相似的建议。但是，是Linsker使用植根于信息理论的抽象概念提出了最大互信息(infomax)原理。Linsker的文章重新激发了把信息理论应用到神经网络中的兴趣。特别是，Bell and Sejnowski(1995)所作的信息理论对盲信号源分离问题的应用已经促使许多研究者探索用于求解统称为盲反卷积的很大一类问题的其他信息理论模型。

同样在1988年，Broomhead和Lowe描述了使用径向基函数(radial basis function，RBF)设计多层前馈网络的过程，RBF提供了多层感知器的另一选择。径向基函数的基本想法至少追溯到Bashkirov，Braverman and Muchnik(1964)首先提出的势函数方法以及Aizerman，Braverman

and Rozonoer(1964a,b)发展的势函数理论。Duda 和 Hart(1973)的经典著作《*Pattern Classification and Scene Analysis*》(模式分类和场景分析)给出了势函数方法的一个描述。不过，Broomhead and Lowe 的文章导致了联系神经网络设计和数值分析的中重要领域以及线性自适应滤波器的大量研究工作。1990 年 Poggio and Girosi(1990a)利用 Tikhonov 的正则化理论进一步丰富了 RBF 网络理论。

1989 年 Mead 的《*Analog VLSI and Neural Systems*》(模拟 VLSI 和神经系统)一书出版。这本书把从神经生物学和 VLSI 技术吸取的概念进行了不寻常的融合。最重要的是，它包括 Mead 和他的合作者写的关于硅视网膜和硅耳蜗的几章，这些都是 Mead 创造性思维的活生生的例子。

在 20 世纪 90 年代早期，Vapnik 和他的合作者发明了具有强大计算能力的一种有监督学习网络称为*支持向量机*(support vector machine,SVM)，用于解决模式识别、回归和密度估计等问题(Boser,Guyon and Vapnik,1992;Cortes and Vapnik,1995;Vapnik,1995,1998)。这种新方法是基于有限样本学习理论的结果。支持向量机的一个新颖的特征就是在它们的设计中以自然的方式包含了 Vapnik-Chervonenkis(VC)*维数*。VC 维数提供了神经网络从一个样本集中学习能力的一种度量(Vapnik and Chervonenkis,1971;Vapnik,1982)。

现在已很好地建立了*混沌*是构成物理现象的关键方面。许多人提出了一个问题：在神经网络研究中混沌起关键作用吗？在生物环境下 Freeman(1995) 相信这个问题的答案是肯定的。根据 Freeman 的看法，神经活动的模式不是从脑外部强加的，而是从内部构建的。特别是，混沌动力学对神经元群体的内部和它们之间出现自组织模式需要的条件提供了进行描述的一个基础。

也许对 20 世纪 80 年代神经网络兴趣的复兴最有影响的是 Hopfield 1982 年的文章和 Rumelhart 和 McLelland 1986 年的两卷书，而不是其他的著作。神经网络从 McCulloch 和 Pitts 的早期岁月算起当然已走过了很长一段路。确实它们已确立了它们作为植根于神经科学、心理学、数学、物理学和工程的交叉学科的地位。无需赘言，现在它们确立了这样的地位并将在理论、设计和应用上继续深入。 44

注释和参考文献

[1] 这个神经网络的定义来自 Aleksander and Morton (1990)。

[2] 神经网络侧重于神经建模、认知和神经生理学方面的补充材料参看 Anderson (1995)。有关脑计算方面易读的材料可参看 Churchland and Sejnowski (1992)。有关神经机制和人脑的更详细的描述可参看 Kandel and Schwartz (1991)，Shepherd (1990 a,b)，Koch and Segev (1989)，Kuffler et al. (1984)和 Freeman (1975)。

[3] 关于 sigmoid 函数和相关问题全面叙述可参看 Menon et al. (1996)。

[4] logistic 函数或更精确地说 logistic 分布函数的命名，来自见于大量文献的深奥的"logistic 增长律"。利用适当的度量单位，假定所有的增长过程可表示为 logistic 分布函数

$$F(t) = \frac{1}{1 + e^{\alpha t - \beta}}$$

其中 t 代表时间，α，β 为常数。但是结果证明不仅是 logistic 分布，而且 Gauss 分布和其他分布都能应用于相同的数据，取得一样或更好的拟合(Feller，1968)。

[5] 根据 Kuffler et al. (1984)，"接受域"(receptive field)这个术语最早由 Sherrington (1906)创

造的，并被 Hartline (1940)重新引入。在视觉系统环境下，神经元的接受域是指视网膜曲面上限制为光引起神经元放电的区域。

[6] 看来权值共享技术最早在 Rumelhart et al. (1986b)中描述。

[7] 这里给出的历史注释大部分(但不是全部)基于下列资料：(1)Saarinen et al.(1992)的文章；(2)Rall (1990)的章节；(3)Widrow and Lehr (1990)的文章；(4)Cowan (1990)以及 Cowan and Sharp (1988)的文章；(5)Grossberg (1988c)的文章；(6)关于神经计算的两卷书(Anderson et al., 1990; Anderson and Rosenfeld, 1988)；(7)Selfridge et al.(1988)的章节；(8)von Neumann 关于计算和计算机理论的论文集(Aspray and Burks, 1986)；(9) Arbib (1995)编辑的脑理论和神经网络的手册；(10)Russell and Norvig (1995)的第1章；(11)Taylor (1997)的文章。

习题

神经元模型

1.1 一个 logistic 函数的例子定义为

$$\varphi(v)=\frac{1}{1+\exp(-av)}$$

45 它的极限值为0和1。证明它关于 v 的导数由

$$\frac{d\varphi}{dv}=a\varphi(v)[1-\varphi(v)]$$

给出。这个导数在原点的值是多少？

1.2 一个奇 sigmoid 函数定义为

$$\varphi(v)=\frac{1-\exp(-av)}{1+\exp(-av)}=\tanh\left(\frac{av}{2}\right)$$

其中 tanh 代表双曲正切。这第二个 sigmoid 函数的极限值为 -1 和 +1。证明 $\varphi(v)$关于 v 的导数由

$$\frac{d\varphi}{dv}=\frac{a}{2}[1-\varphi^2(v)]$$

给出。这个导数在原点的值是多少？假设倾斜参数 a 无穷大，$\varphi(v)$的结果是什么形式？

1.3 另外一个奇 sigmoid 函数是代数 sigmoid：

$$\varphi(v)=\frac{v}{\sqrt{1+v^2}}$$

它的极限值为 -1 和 +1。证明它关于 v 的导数由

$$\frac{d\varphi}{dv}=\frac{\varphi^3(v)}{v^3}$$

给出。这个导数在原点的值是多少？

1.4 考虑下列两个函数：

(i) $\varphi(v)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{v}\exp\left(-\frac{x^2}{2}\right)dx$

(ii) $\varphi(v)=\frac{2}{\pi}\tan^{-1}(v)$

解释为什么两个函数都满足 sigmoid 函数的要求。怎样区别它们?

1.5 在问题 1.1 至问题 1.4 的五个 sigmoid 函数中哪些是累积(概率)分布函数? 证明你的答案的正确性。

1.6 考虑图 1-26 所示的拟线性激活函数 $\varphi(v)$。

(a)写出 $\varphi(v)$关于 v 的函数公式。

(b)若 a 允许趋于 0，$\varphi(v)$会出现什么情况?

1.7 关于图 1-27 所示的拟线性激活函数 $\varphi(v)$重复习题 1.6。

46

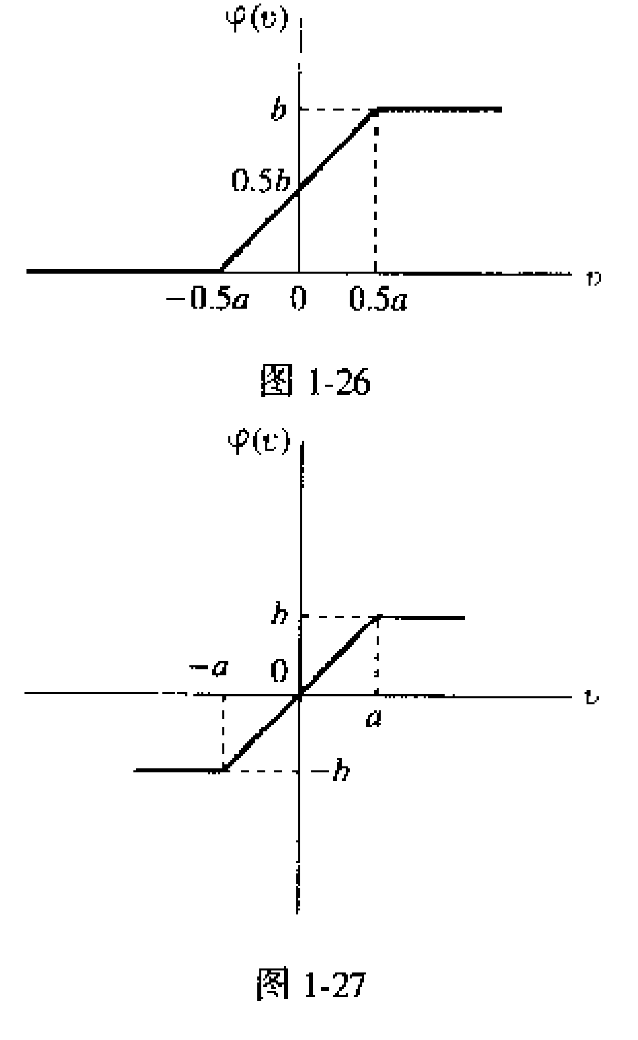


图 1-26

图 1-27

1.8 一个神经元具有问题 1.1 的 logistic 函数定义的激活函数 $\varphi(v)$，其中 v 是诱导局部域并且倾斜参数 a 可调节。令 x_1，x_2，…，x_m 为作用于神经元源节点的输入信号，b 表示偏置。为了表示方便起见，我们将吸收倾斜参数 a 到诱导局部域 v，写成

$$\varphi(v)=\frac{1}{1+\exp(-v)}$$

你将如何改变输入 x_1，x_2，…，x_m 产生和以前一样的结果? 证明你的回答的正确性。

1.9 神经元 j 从其他四个神经元接受输入，它们的活动性级别为 10，-20，4 和 -2。神经元 j 的每个突触权值分别为 0.8，0.2，-1.0 和 -0.9。计算下列两种情况下神经元 j 的输出:

(a)神经元是线性的。

(b)神经元由 McCulloch-Pitts 模型表示。

假设神经元的偏置为 0。

1.10 对基于 logistic 函数

$$\varphi(v)=\frac{1}{1+\exp(-v)}$$

的神经元模型重复问题 1.9。

1.11 (a)证明神经元的 McCulloch-Pitts 形式模型可由 sigmoid 神经元逼近(即利用具有非常大的突触权值的 sigmoid 激活函数的神经元)。

(b)证明线性神经元可由具有很小突触权值的 sigmoid 神经元逼近。

网络结构

1.12 一个全连接的前馈网络具有 10 个源节点，2 个隐层，一个隐层有 4 个神经元，另一个有 3 个神经元，以及 1 个输出神经元。构造这个网络的结构图。

1.13 (a)图 1-28 表示一个 2-2-2-1 前馈网络的信号流图。函数 $\varphi(\cdot)$表示 logistic 函数。写出由这个网络定义的输入输出映射。

47

(b)假设图 1-28 信号流图的输出神经元运行在它的线性区域。写出由这个网络定义的输入输出映射。

1.14 图 1-28 所描述的神经网络没有偏置。假设第一隐层的顶和底神经元的偏置分别为 -1 和 +1，第二隐层的顶和底神经元的偏置分别为 +1 和 -2。写出由这个网络定义输入

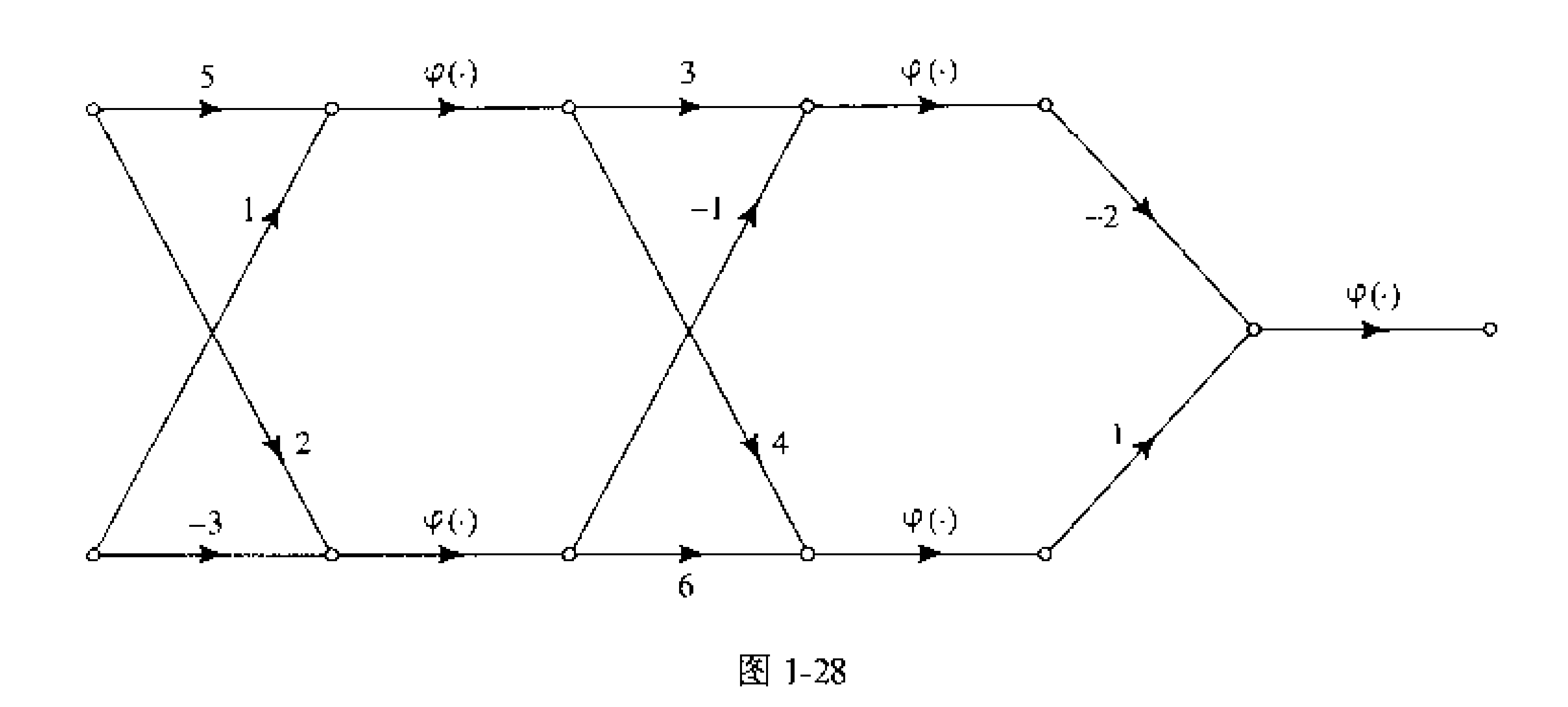


图 1-28

输出映射的新形式。

1.15 考虑一个多层前馈网络，它所有的神经元运行在它们的线性区域。证明这样的网络等价于单层前馈网络的结论。

1.16 构造一个全连接的递归网络，它具有5个神经元，但没有自反馈。

1.17 图1-29表示两个神经元的递归网络信号流图。写出定义 $x_1(n)$ 和 $x_2(n)$ 演变的非线性差分方程。这两个变量分别定义顶部和底部神经元的输出。这个方程的阶是多少?

1.18 图1-30表示具有自反馈的两个神经元的递归网络信号流图。写出描述系统运行的两个一阶耦合非线性差分方程组。

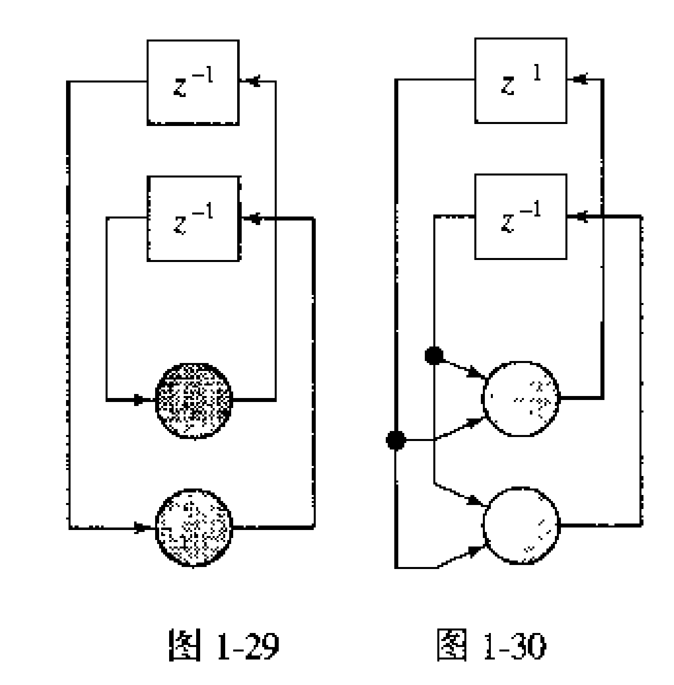


图 1-29　　图 1-30

48

1.19 一个递归网络具有3个源节点、2个隐藏神经元和4个输出神经元。构造描述这样一个网络的结构图。

知识表示

1.20 一个有用的预处理形式是基于由差分方程(用于实数值数据)

$$y(n) = w_1 y(n-1) + w_2 y(n-2) + \cdots + w_M y(n-M) + \nu(n)$$

描述的自回归(AR)模型，其中 $y(n)$ 是模型输出，$\nu(n)$ 为从零均值和预定方差的白噪声过程抽取的样本，$w_1, w_2, \cdots, w_M$ 是AR模型的系数，而 M 为模型阶数。证明利用这个模型提供两种形式的几何不变性：(a)尺度大小；(b)时间平移。在神经网络中怎样利用这两种不变性?

1.21 令 $\mathbf{x}$ 为输入向量，$\mathbf{s}(\alpha,\mathbf{x})$ 为依赖于参数 α 的作用于 $\mathbf{x}$ 的变换算子。它满足两个要求：

49

- $\mathbf{s}(0,\mathbf{x}) = \mathbf{x}$
- $\mathbf{s}(\alpha,\mathbf{x})$ 关于 α 可微

切向量定义为偏导数 $\partial\mathbf{s}(\alpha,\mathbf{x})/\partial\alpha$(Simard et al., 1992)。

假设 $\mathbf{x}$ 代表一幅图像，α 是旋转参数。在 α 很小时你怎样计算切向量? 切向量关于原图像的旋转是局部不变的，为什么?

第2章 学习过程

2.1 简介

对于神经网络具有首要意义的性质是网络能从环境中学习的能力，并通过学习改善其行为。对行为的改善是随时间依据某一规定的度量进行的。神经网络通过施加于它的突触权值和偏置水平的调节的交互过程来学习它的环境。理想情况下，神经网络在每一次重复学习过程后对它的环境便有更多的了解。

有过多的与“学习”这个概念相联系的行为，以至不能以精确的方式对其定义。而且，学习过程是这样一种观点问题，使得在对这个术语的精确定义上很难达成一致。比如，心理学家眼中的学习与课堂中的学习是截然不同的。需认识我们的特殊兴趣在于神经网络，我们使用一个从 Mendel and McClaren(1970)修改过的一个关于学习的定义。

我们在神经网络的背景中定义学习如下：

学习是一个过程，通过这个过程神经网络的自由参数在其嵌入的环境的激励过程之下得到调节。学习的类型由参数改变的方式决定。

这个学习过程的定义隐含着如下的事实：

1. 神经网络被一个环境所激励。
2. 作为这个激励的结果，神经网络在它的自由参数上发生变化。
3. 由于神经网络内部结构的改变而以新的方式响应环境。

建议解决学习问题的一个恰当定义的规则集合称作*学习算法*[1]。就像人们预料的那样，对于神经网络的设计没有惟一的学习算法。然而，我们有由不同学习算法表示的一组工具，每一个有它自己的优势。基本上，学习算法在其对神经元的突触权值的调节方式各不相同。 50 要考虑的另一方面是由一组相互连接的神经元组成神经网络（学习机器）与其环境联系的方式。从后一个方面说，我们提到*学习范例*是指神经网络运行于其中的环境的一个模型。

本章的组织

本章由四个相互联系的部分组成。第一部分包括第 2.2 节到 2.6 节，我们讨论五个基本的学习算法：误差 - 修正学习，基于记忆的学习，Hebb 学习，竞争学习和 Boltzmann 学习。误差修正学习植根于最优滤波。基于记忆的学习通过明确地记住训练数据来进行。Hebb 学习和竞争学习都是受了神经生物学上的考虑的启发。Boltzmann 学习有所不同，因为它是建立在从统计学力学借来的思想基础上。

本章的第二部分探讨学习范例。2.7 节讨论信任赋值问题，它是学习过程的基础。2.8 节和 2.9 节概述两个基本学习范例：(1)有教师学习，(2)无教师学习。

本章的第三部分包括 2.10 节到 2.12 节，考察学习任务、记忆和自适应的问题。

本章的最后部分包括 2.13 节到 2.15 节，处理学习过程的概率和统计方面。2.13 节讨论

偏置/方差闲境。2.14节讨论基于VC维数概念的统计学习理论，VC维数提供了对机器能力的一个测量方法。2.14节介绍另一个重要概念：可能近似正确(PAC)学习，它为学习过程提供一个保守的模型。

本章在2.16节中用一些最后的评述作为结束。

2.2 误差修正学习

为了说明第一条学习规则，考虑如图2-1a所示由一个神经元 k 构成前馈神经网络输出层的惟一计算节点的简单情况。神经元 k 被一层或多层隐藏神经元产生的信号向量 $\mathbf{x}(n)$ 驱动，这些隐藏神经元自身由作用于神经网络的源节点(也就是输入层)的输入向量驱动。参数 n 表示离散时间，或者更确切地说，是调节神经元 k 的突触权值的交互过程的时间步。神经元 k 的输出信号由 $y_k(n)$ 表示。这个描述神经网络惟一输出的输出信号与由 $d_k(n)$ 表示的期望响应或目标输出比较。由此产生由 $e_k(n)$ 表示的误差信号。由定义，我们有

$$e_k(n) = d_k(n) - y_k(n) \tag{2.1}$$

误差信号 $e_k(n)$ 驱动控制机制，其目的是将修正调节序列作用于神经元 k 的突触权值。修正调节能够以一步步逼近的方式使输出信号 $y_k(n)$ 向期望输出 $d_k(n)$ 靠近。这一目标通过最小化代价函数或性能指标 $\mathscr{E}(n)$ 来实现。$\mathscr{E}(n)$ 借助误差信号 $e_k(n)$ 定义如下：

51

$$\mathscr{E}(n) = \frac{1}{2}e_k^2(n) \tag{2.2}$$

也就是说，$\mathscr{E}(n)$ 是误差能量的瞬时值。这种对神经元 k 的突触权值步步逼近的调节将持续下去，直到系统达到稳定状态(即突触权值基本稳定下来)。这时，学习过程终止。

在这里，描述的学习过程显然应被称为误差－修正学习。特别，对代价函数 $\mathscr{E}(n)$ 的最小化导致了通常被称作增量规则或Widrow-Hoff规则的学习规则，规则的命名是为了纪念它的发明者(Widrow and Hoff, 1960)。令 $w_{kj}(n)$ 表示在第 n 时间步，被信号向量 $\mathbf{x}(n)$ 的 $x_j(n)$ 分量激发的神经元 k 的突触权值。根据增量规则，在第 n 时间步作用于突触权值的调节量 $\Delta w_{kj}(n)$ 定义如下：

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n) \tag{2.3}$$

这里 η 是一个正的常量，它决定学习过程中从一步到另一步时的学习率。所以，我们自然而然地称 η 为学习率参数。换言之，增量规则可以表述为：

52

作用于神经元突触权值的调节量正比于本次学习中误差信号与突触的输入信号的乘积。

牢记这里表述的增量规则假定误差信号是直接可测量的。为了这样的测量是可行的，我们显然需要与神经元 k 直接相连的外部源提供期望响应。换言之，神经元 k 对外部世界是可见的，如图2-1a所示。从该图中还可以看到，误差－修正学习实际上带有局部性质。这仅仅是说出增量规则计算的突触调节局部于神经元 k 周围。

在计算突触调节量 $\Delta w_{kj}(n)$ 后，突触权值 w_{kj} 的更新值由

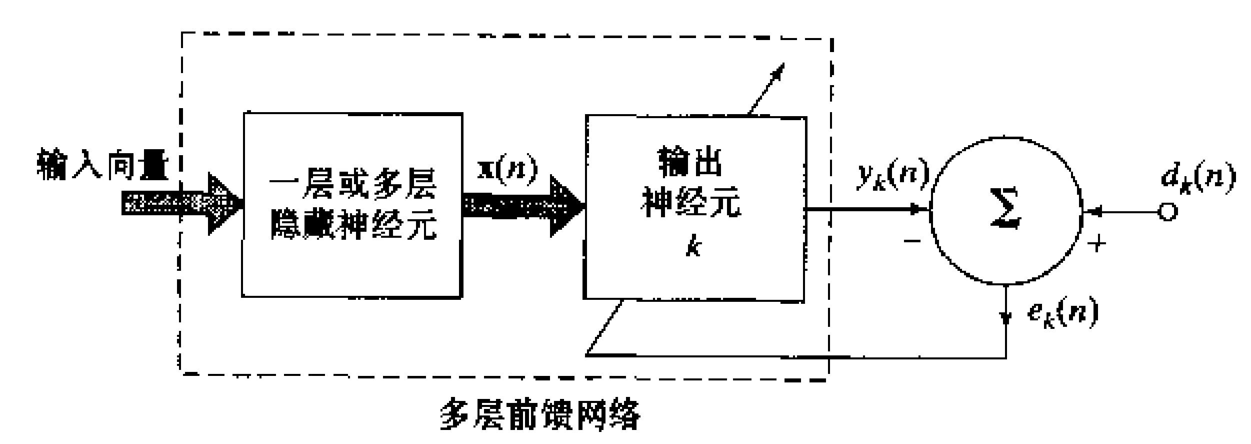
$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n) \tag{2.4}$$

确定。实际上，$w_{kj}(n)$ 和 $w_{kj}(n+1)$ 可以分别被视为突触权值 w_{kj} 的旧值和新值。从计算的角度，我们也可写为：

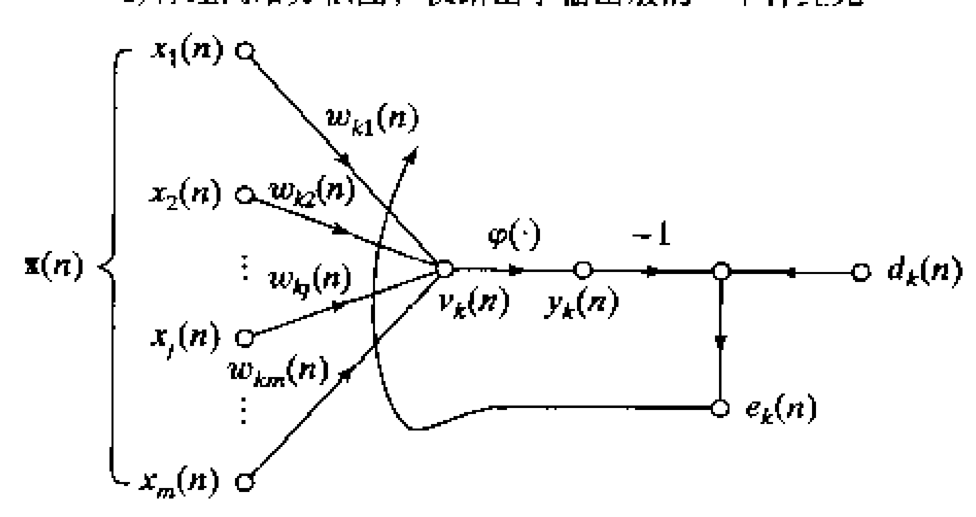
$$w_{kj}(n) = z^{-1}[w_{kj}(n+1)] \tag{2.5}$$

这里 z^{-1} 是单元 - 延迟操作符。也就是说，z^{-1} 表示一个存储元件。

图 2-1b 用信号流图表示误差 - 修正的学习过程，其焦点集中在神经元 k 周围的活动。输入信号 x_j 和神经元 k 的诱导局部域 v_k 分别称作神经元 k 的第 j 个突触的前突触信号和后突触信号。从图 2-1b 看出误差-修正学习是闭环反馈系统的一个例子。由控制论我们知道这种系统的稳定性由构成系统的反馈环路的参数决定。在这里，我们仅有一个单一反馈环路，具有特别意义的参数之一是学习率参数 η。因此，仔细选取 η 以取得重复学习过程的稳定性或收敛性是很重要的。对 η 的选择对学习过程的准确性及其他方面也有深刻的影响。简言之，学习率参数 η 在实际决定误差 - 修正学习性能时起着关键作用。



a)神经网络方框图，仅给出了输出层的一个神经元



b)输出神经元信号流图

图 2-1 误差 - 修正学习图示

误差 - 修正学习将在第 3 章和第 4 章详细论述，第 3 章讨论单层前馈网络，第 4 章详细论述多层前馈网络。

2.3 基于记忆的学习

在基于记忆的学习中，所有(或大部分)以往的经验被显式地存储到正确分类的输入 - 输出实例 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ 的大量记忆中，这里 $\mathbf{x}_i$ 表示输入向量，d_i 表示对应的期望响应。不失一般性，我们限制期望响应为一个标量。例如，在二值模式分类中，考虑有两个分别表示为 $\mathscr{C}_1$ 或 $\mathscr{C}_2$ 的类别/假设。在这个例子中，期望响应 d_i 对类 $\mathscr{C}_1$ 取值 0(或 -1)，对类 $\mathscr{C}_2$ 取值 1。当需要对测试向量 $\mathbf{x}_{test}$(以前未见过)进行分类时，算法通过提取并分析 $\mathbf{x}_{test}$ 的局部邻域中的训练数据进行响应。

53

所有基于记忆的学习算法包括两个重要的组成部分：

- 用于定义测试向量 $\mathbf{x}_{test}$ 的局部邻域的准则。
- 用于 $\mathbf{x}_{test}$ 的局部邻域中的训练实例的学习规则。

算法随这两个组成部分的不同而不同。

在一个简单而有效的称作最近邻规则[2]的基于记忆的学习类型中，局部邻域被定义为测试向量 $\mathbf{x}_{test}$ 的直接邻域的训练实例。特别，向量

$$\mathbf{x}'_N \in \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \tag{2.6}$$

被称作 $\mathbf{x}_{test}$ 的最近邻，如果

$$\min_i d(\mathbf{x}_i, \mathbf{x}_{test}) = d(\mathbf{x}'_N, \mathbf{x}_{test}) \tag{2.7}$$

这里，$d(\mathbf{x}_i, \mathbf{x}_{test})$ 是向量 $\mathbf{x}_i$ 和 $\mathbf{x}_{test}$ 的欧几里德距离。与最短距离相关联的类别，也就是向量 $\mathbf{x}'_N$ 被划分的类别。这个规则独立于产生训练实例的基本分布。

Cover and Hart(1967)形式地研究了作为一个模式分类工具的最近邻规则。在那里提出的分析基于两个假设：

- 分类实例 $(\mathbf{x}_i, d_i)$ 按照实例 $(\mathbf{x}, d)$ 的联合概率分布是独立同分布的(iid)。
- 样本大小 N 是无限大的。

在这两个假设下，可以证明，由最近邻规则引起的分类误差概率被限制在贝叶斯误差概率(也就是所有判定规则中的最小误差概率)的两倍以上。贝叶斯误差概率在第3章讨论。在这个意义上，可以说，无限大小的训练集中有一半分类信息包含在最近邻中，这是令人惊奇的结果。

最近邻分类器的一个变种是 k－最近邻分类器，它操作如下：

- 对于某一整数 k，确定与测试向量 $\mathbf{x}_{test}$ 最邻近的 k 个类别模式。
- 将 $\mathbf{x}_{test}$ 的 k 个最近邻中出现最多的类别(假设)分配给 $\mathbf{x}_{test}$(即用多数表决进行分类)。

这样，k－最近邻分类器的作用就像一个平均仪器。特别的，对于 $k=3$，k－最近邻分类器鉴别单个的例外(outlier)，如图2-2所示。一个例外是一个观察，这个观察对于我们感兴趣的指定模型是异常大。

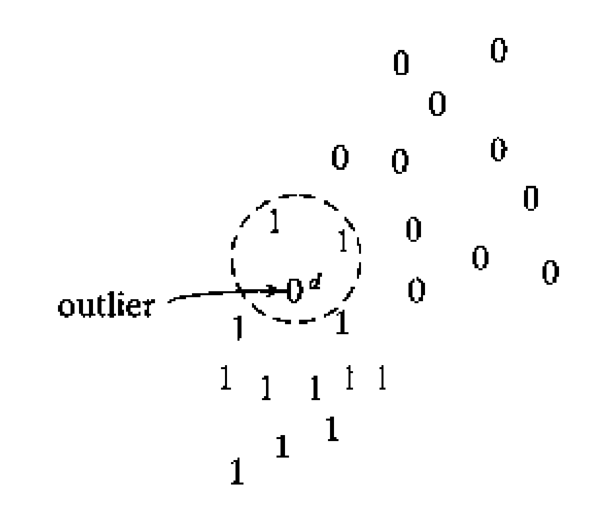


图2-2 分类的例外

虚线圆圈里面的区域包括两个属于分类1的点和一个来自分类0的例外。点 d 对应于测试向量 $\mathbf{x}_{test}$。当 $k=3$，k－最近邻分类器给点 d 指定类别1，即使它与那个例外离得最近

54

在第5章我们讨论另一个重要的称作径向基函数网络的基于记忆的分类器类型。

2.4 Hebb 学习

学习的 Hebb 假设是所有学习规则中最悠久最著名的；它是为了纪念神经心理学家 Hebb (1949)而命名的。下面一段引自 Hebb 的《行为的组织》一书(1949, p.62)：

> 当细胞 A 的一个轴突足够近地刺激细胞 B 并反复或持续地激励它时，某种增长过程或新陈代谢变化在一个或两个细胞中发生，这使得 A 作为激励 B 的细胞中的一个的效率被增大。

Hebb 提出将这个变化作为联想学习的基础(在细胞水平上)，其结果是按空间分布的"神经细胞集合"的活动模式的持续修改。

这个陈述是在神经生物学的背景中做出的。我们可以将之扩充并重述为二分规则(Stent, 1973; Changeux and Danchin, 1976)：

1. 如果在突触(连接)每一边的两个神经元被同时(即同步)激活，那么那个突触的强度被选择性地增强。

2. 如果在突触每一边的两个神经元被异步激活，那么那个突触被选择性地减弱或消除。

这样的突触被称作 Hebb 突触[3]。(最初的 Hebb 规则不包括第二部分)。更确切地说，我们定义 Hebb 突触为这样的一个突触，它使用一个*依赖时间的、高度局部的和强烈交互的机制来提高突触效率作为前突触和后突触活动间的相互关系的一个函数*。从这个定义，我们可以得出下面标志 Hebb 突触特征的 4 个重要机制(特性)：

1. *时间依赖机制*。这一机制是指这样一个事实，Hebb 突触中的修改取决于前突触和后突触信号出现的确切时间。

2. *局部机制*。突触在其本质上是传输的场所，其中信息－承载信号(表示了前突触和后突触单元中正在进行的活动)处于时空的邻近。Hebb 突触利用这个局部可用信息产生由输入确定的局部突触修改。 55

3. *交互机制*。Hebb 突触中改变的发生取决于突触两边的信号。也就是说，Hebb 学习的方式，在我们无法从这两个活动中任意一个自身作出预测的意义上说，是取决于前突触和后突触信号间的"真正交互"。注意这个依赖或交互可能本质上是确定性或随机性的。

4. *关联或相关机制*。对 Hebb 学习假设的解释之一是突触效率的改变条件为前后突触信号的关联。于是，根据这种解释，前突触和后突触信号的同时发生(有一个短的时间间隔)足以产生对突触的修改。正是由于这个原因，Hebb 突触又被称作*关联突触*。在对 Hebb 学习假设的另一种解释中，我们可以从统计学的角度考虑作为 Hebb 突触特征的交互机制。特别，前突触和后突触信号在时间上的相关被认为决定着突触的变化。所以，Hebb 突触也被称作*相关突触*。相关确实是学习的基础(Eggermont, 1990)。

突触的增强和抑制

这里表述的 Hebb 突触定义不包括那些可能导致连接着一对神经元的突触减弱的附加过程。确实，我们可以通过认识正相关活动导致突触增强和非相关或负相关活动导致突触减弱来推广 Hebb 修改的概念(Stent, 1973)。突触抑制也可以是非交互类型的。特别是，突触减弱的交互条件可能仅仅是前突触或后突触活动的不一致。

我们更进一步，将突触修改分为 Hebb *式*、*反* － Hebb *式*和*非* － Hebb *式*(Palm, 1982)。按照这种划分，Hebb 突触的强度因为正相关的前突触和后突触信号而增加，以及当信号或者是不相关或者是负相关的而降低强度。相反，反－Hebb 突触由正相关的前突触和后突触信号而减弱，因负相关的信号而增强。然而，在 Hebb 突触和反－Hebb 突触两者中，对突触效率的修改依赖于在本质上是依赖时间的、高度局部的和强烈交互的机制。在那种意义下，反 Hebb 突触的性质仍然是 Hebb 式的，尽管不是在功能上。另一方面，非－Hebb 突触不包含 Hebb 机制中的任何一种。

Hebb 修改的数学模型

为了从数学角度阐明 Hebb 学习，考虑神经元 k 的一个突触权值 w_{kj}，分别用 x_j 和 y_k 表示前突触和后突触信号。在时间步 n 用于突触权值 w_{kj} 的调整用一般化形式如下：

56

$$\Delta w_{kj}(n) = F(y_k(n), x_j(n)) \tag{2.8}$$

表示，其中 $F(\cdot,\cdot)$是后突触和前突触信号的函数。信号 $x_j(n)$和 $y_k(n)$经常被当做是没有维数的。公式(2.8)允许有多种形式，所有这些形式都称为是 Hebb 形式。下面，我们考虑两种这样的形式。

Hebb 假设　Hebb 学习的最简单形式描述为

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n) \tag{2.9}$$

其中 η 是决定学习率的正值常量。式(2.9)清楚地强调了 Hebb 突触的相关性质。它有时被称作*活动产生规则*。图 2-3 中上方的曲线显示式(2.9)中改变量 Δw_{kj} 随输出信号(后突触活动)y_k 改变的图形表示。从这个表示中，我们看出重复使用输入信号(前突触活动)x_j 将导致的 y_k 增长以及由此引发的指数增长，这将使突触连接进入饱和状态。这时，没有任何信息存储在突触中并且失去选择性。

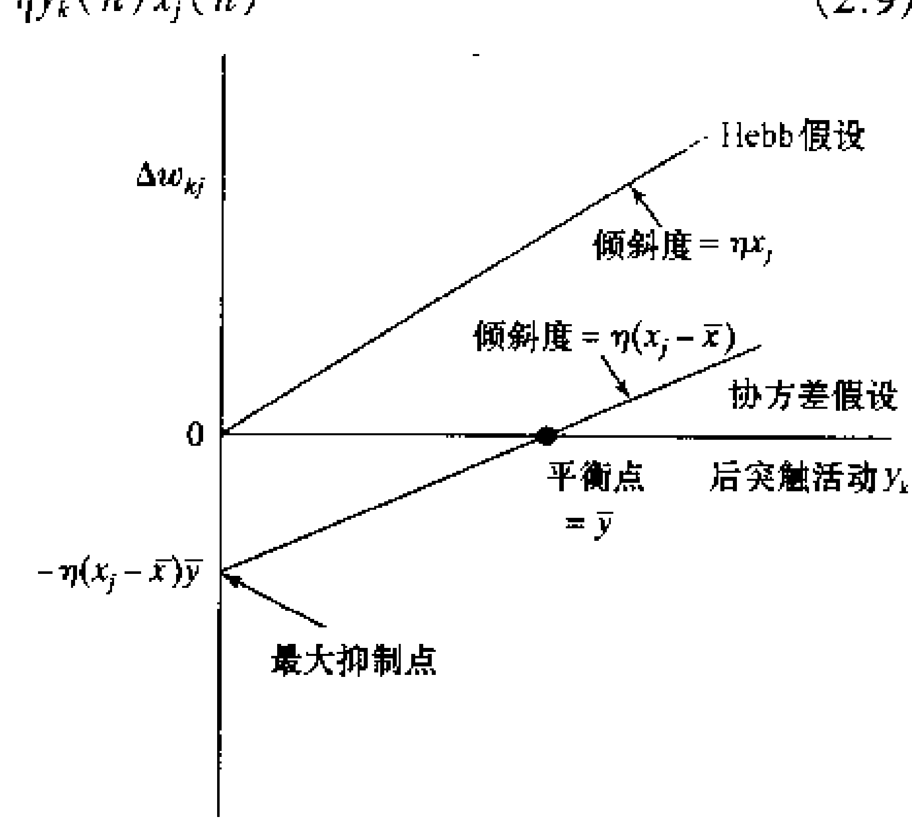


图 2-3　Hebb 假设和协方差假设的图示

协方差假设　克服 Hebb 假设限制的途径之一是使用 Sejnowski(1977a, b)引入的*协方差假设*。在这个假设里，式(2.9)中前突触和后突触信号分别用前突触和后突触信号与它们各自的在一定时间间隔上的期望均值的偏移量所代替。令 $\bar{x}$ 和 $\bar{y}$ 分别表示前突触 x_j 和后突触信号 y_k 的时间 - 均值。按照协方差假设，作用于突触权值 w_{kj} 的调整定义为

57

$$\Delta w_{kj} = \eta(x_j - \bar{x})(y_k - \bar{y}) \tag{2.10}$$

其中 η 是学习率参数。x 和 y 的均值构成前突触和后突触阈值，它决定突触修改的正负值。特别，协方差假设考虑了下述方面：

- 收敛于非平凡状态，当 $x_k = x$ 或 $y_j = \bar{y}$ 时到达。
- 对突触加强(即增加突触强度)和突触抑制(即降低突触强度)两者的预测。

图 2-3 说明 Hebb 假设和协方差假设之间的差别。在两种情况下，Δw_{kj} 对 y_k 的依赖是线性的；然而，在 Hebb 假设中与 y_k 轴的相交是在原点，而在协方差假设中是在 $y_k = y$ 处。

我们从式(2.10)得出如下重要观察：

1. 如果有足够的前突触和后突触活动程度，也就是同时满足条件 $x_j > \bar{x}$ 和 $y_k > \bar{y}$，则突触权值 w_{kj} 得到加强。

2. 如果至少满足下条件任意之一，则突触权值被减弱：

- 在缺乏足够的后突触激活(即 $y_k < \bar{y}$)的条件下前突触激活(即 $x_j > \bar{x}$)。

- 在缺乏足够的前突触激活(即 $x_j < \bar{x}$)的条件下后突触激活(即 $y_k > \bar{y}$)。

这种行为可以被认为是输入模式间时间竞争的一种形式。

在称作海马区的脑区域提供了对 Hebb 学习有力的生理学证据[4]。海马区在学习或记忆的某些方面起着重要作用，这种生理学证据使得 Hebb 学习更具吸引力。

2.5 竞争学习

顾名思义，在竞争学习[5]中，神经网络中的输出神经元彼此通过竞争来成为活跃的(点火)。在基于 Hebb 学习的神经元网络里，若干输出神经元可能同时处于激活状态，而在竞争学习里，在任意时刻只有一个输出神经元是激活的。正是这个特性使竞争学习高度适合于发现统计上的突出特征，这些特征可以用来分类输入模式的集合。

对于竞争学习规则，有三个基本元素(Rumelhart and Zipser,1985)：

- 一个神经元集合，这些神经元除了一些随机分布的突触权值之外是完全相同的，并且由于突触权值的不同而对一个给定的输入模式集合有不同的响应。
- 对每个神经元的强度加上的限制。
- 允许神经元为响应一个给定输入子集的权利而竞争的机制，从而使得每次只有一个输出神经元或者每组只有一个神经元是激活的(即"开")。竞争获胜神经元被称为*胜者全得*(winner-takes-all)神经元。

因此，网络的神经元个体学会专门辨别相似模式的总体；这样做的结果，它们成为不同类别输入模式的特征探测器。 58

在最简单的竞争学习形式中，神经网络有单一的一层输出神经元，其中的每一个都与输入节点完全连接。网络可以包含神经元的反馈连接，如图 2-4 所示。在这里描绘的网络结构中，反馈连接执行侧向抑制[6]，每个神经元都试图抑制与其侧向连接的神经元。相反，图 2-4 的网络结构中的所有前馈突触连接都是激活的(兴奋的)。

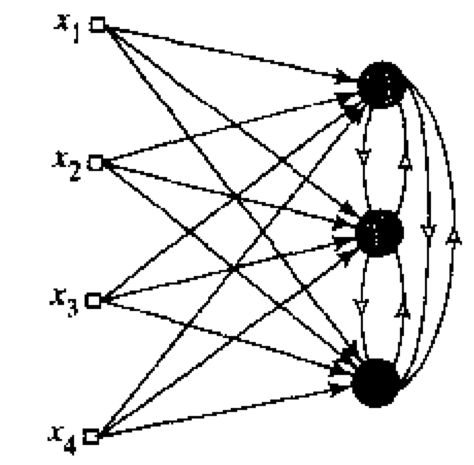


图 2-4 一个简单竞争学习网络的结构图，它具有从源节点到神经元的前馈(兴奋的)连接和神经元之间的侧向(抑制的)连接(侧向连接由空心箭头标示出)

对于一个要想成为获胜神经元的神经元 k，对于指定输入模式 $\mathbf{x}$ 的诱导局部域 v_k 必需是网络结构中所有神经元中最大的。获胜神经元 k 的输出信号 y_k 被置为 1；竞争失败的所有神经元的输出信号被置为 0。这样，我们有

$$y_k = \begin{cases} 1, & \text{如果 } v_k > v_j \text{ 对于所有 } j, j \neq k \\ 0, & \text{否则} \end{cases} \quad (2.11)$$

其中，诱导局部域 v_k 表示结合所有到达神经元 k 的前向和反馈输入的动作。

令 w_{kj} 表示连接输入节点 j 到神经元 k 的突触权值。假定每个神经元被分配(allotted)固定量的突触权值(即所有突触权值都是正的)，权值分布在它的输入节点之中；也就是

$$\sum_j w_{kj} = 1, \quad \text{对于所有 } k \quad (2.12)$$

然后神经元通过将突触权值从它的不活跃输入移向活跃输入来进行学习。如果神经元对一个特定输入模式不响应，那么没有学习发生在那个神经元上。如果一个特定神经元赢得了竞

争，这个神经元的每个输入节点以一定比例释放它的突触权值，释放的权值然后平均分布到活跃输入节点上。按照标准的竞争学习规则，作用于突触权值 w_{kj} 的改变量 Δw_{kj} 定义为

$$\Delta w_{kj} = \begin{cases} \eta(x_j - w_{kj}) & ,\text{如果神经元 } k \text{ 竞争成功} \\ 0 & ,\text{如果神经元 } k \text{ 竞争失败} \end{cases} \tag{2.13}$$

其中 η 是学习率参数。这个规则具有将获胜神经元 k 的突触权值向量 $\mathbf{w}_k$ 向输入模式 $\mathbf{x}$ 移动

59

的整体效果。

我们可以使用图 2-5 中描绘的几何类比来说明竞争学习的本质(Rumelhart and Zipser, 1985)。假定每个输入模式(向量)$\mathbf{x}$ 具有某一常量欧几里德长度，使得我们可以将它看作是 N - 维单位球上的一个点，其中 N 是输入节点的数目。N 也表示每个突触权值向量 $\mathbf{w}_k$ 的维数。进一步假定网络中所由神经元都被限定具有相等的欧几里德长度(范数)，表示如下：

$$\sum_j w_{kj}^2 = 1, \text{对所有 } k \tag{2.14}$$

当突触权值被适当设定，它们就成为落入同一 N - 维单位球的一组向量。在图 2-5a 中我们显示了三个用点表示的刺激模式的自然分组(簇)。这个图也包括一个可能的网络初始状态(用叉表示)，它可能存在于学习之前。图 2-5b 显示网络作为使用竞争学习结果的一个典型的终止状态。特别，每个输入神经元通过将其突触权值移向簇的重心而发现这以输入模式的簇(Rumelhart and Zipser, 1985; Hertz et al., 1991)。这个图说明了神经网络通过竞争学习进行聚类的能力。然而，为了这一功能能以"稳定的"方式执行，开始时输入模式必需落入充分分离的分组中。否则，网络可能不稳定，因为它将不再以同样的输出神经元响应给定的输入模式。

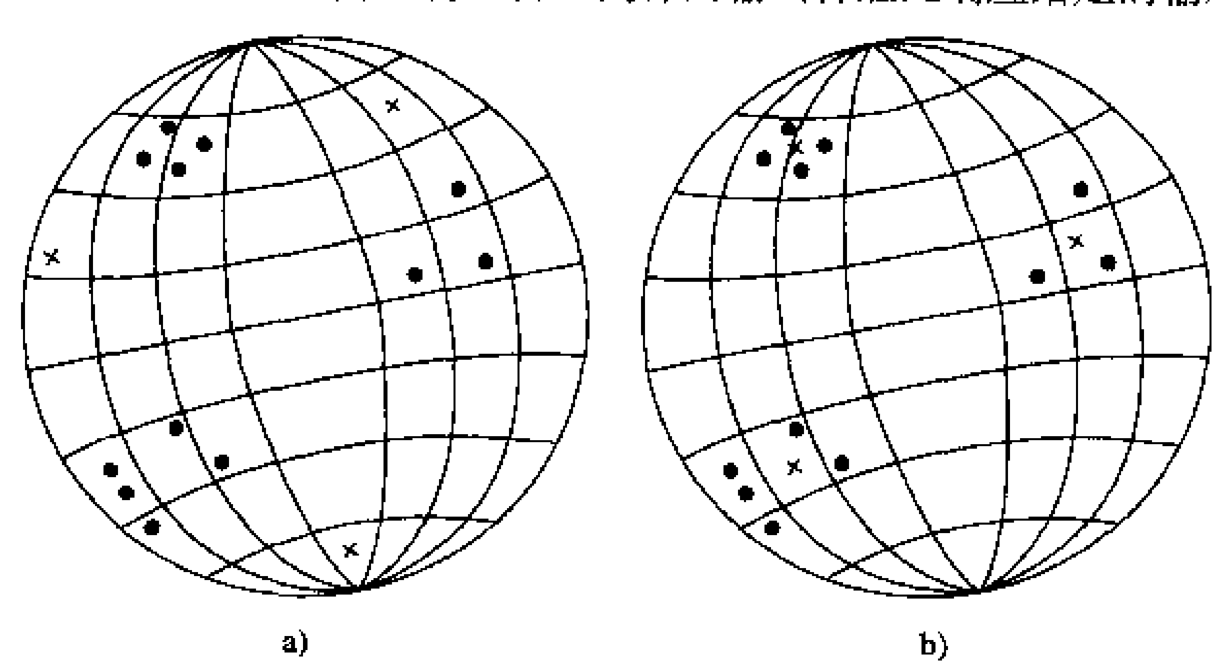


图 2-5 竞争学习过程的几何解释，点代表输入向量，叉代表 3 个输出神经元的突触权值向量
a)网络的初始状态 b)网络的终止状态

2.6 Boltzmann 学习

为了纪念 Ludwig Boltzmann 而命名的 Boltzmann 学习规则是一个从植根于统计力学中的思

60

想推导得出的随机学习算法[7]。基于 Boltzmann 学习规则设计的神经元网络称作 Boltzmann 机(Ackley et al., 1985; Hinton and Sejnowski, 1986)。

在 Boltzmann 机中，神经元构成递归结构，并以二值方式运作，因为，例如它们要么处于用 +1 表示的"开"状态，要么处于用 -1 表示的"关"状态。Boltzmann 机由能量函数 E 所表

征，能量函数的值由机器的个体神经元占据的特定状态所决定，表示成

$$E = -\frac{1}{2}\sum_{j}\sum_{\substack{k \\ j \neq k}} w_{kj} x_k x_j \tag{2.15}$$

其中 x_j 是神经元j 的状态，w_{kj}是连接神经元j 到神经元k 的突触权值。$j \neq k$ 的事实仅仅意味着机器中没有一个神经元有自反馈。机器的运作是通过在学习过程某一步随机地选择一个神经元(例如神经元 k)，然后在某一温度 T 以概率

$$P(x_k \to -x_k) = \frac{1}{1 + \exp(-\Delta E_k / T)} \tag{2.16}$$

将神经元 k 从状态x_k 反转到状态 $-x_k$，其中 ΔE_k 是由这样的反转所导致的能量改变(即机器能量函数的改变量)。注意，T 并非是物理温度，而是第 1 章解释的伪温度。如这一规则被反复使用，机器将达到热平衡。

Boltzmann 机的神经元分为两类功能组：可见的和隐藏的。可见的神经元提供网络和它在其中运作的环境间的接口，而隐藏神经元总是自由运作。有两种运作模式要加以考虑：

- 钳制条件，在这种情形下可见神经元都被钳制到由环境决定的特定状态。
- 自由运行条件，在这种情形下所有神经元(可见的和隐藏的)都允许自由运作。

令 ρ_{kj}^{+} 表示网络在其钳制条件下神经元j 和 k 的状态间的相关量。令 ρ_{kj}^{-} 表示网络在其自由运作条件下神经元j 和 k 的状态间的相关量。两种相关量都是当机器处于热平衡时的所有可能状态的平均。然后，根据 Boltzmann 学习规则，作用于从神经元 j 到神经元 k 的突触权值的改变量由

$$\Delta w_{kj} = \eta(\rho_{kj}^{+} - \rho_{kj}^{-}), \qquad j \neq k \tag{2.17}$$

定义(Hinton and Sejnowski, 1986)，其中 η 是学习率参数。注意 ρ_{kj}^{+} 和ρ_{kj}^{-} 的值都在 -1 和 $+1$ 范围内。

第 11 章给出对统计力学的简单回顾；在那一章，我们还要详尽讨论 Boltzmann 机和其他随机机器。 61

2.7 信任赋值问题

当研究用于分布式系统的学习算法时，考虑信任赋值(credit assignment)，(Minsky, 1961)的问题是有益处的。基本上，信任赋值问题是将导致整体输出的信任和责任分配给每一个由学习机器作出的内部决策及那些对整体输出起作用的决策的问题。(信任赋值问题也被称作装载问题，即将一组给定的训练数据"装载"给网络的自由参数。)

在很多情形下，输出对内部决策的依赖由学习机器采取的一系列动作所调节。换句话说，内部决策影响采取哪些动作，然后这些动作而不是内部决策直接影响整体输出。在这种情形下，我们可将信任赋值问题分解为两个子问题(Sutton, 1984)：

1. 对输出到动作的信任赋值。这被称为时间信任赋值(temporal credit-assignment)问题，因为它涉及应获得信任的动作被实际采取的时刻。

2. 对动作到内部决策的信任赋值。这被称为结构信任赋值(structural credit-assignment)问题，因为它涉及对系统生成动作的内部结构进行信任赋值。

在多成分学习机器中，当为了提高整个系统的性能我们必须精确判定系统的哪个特定成

分应该改变它的行为及作何等程度的改变时，这是和结构信任赋值问题相关的。另一方面，当学习机器采取很多动作而导致某些输出并且我们必须判定这些动作中有哪些应对输出负责时，这是和时间信任赋值问题相关的。时间和结构信任赋值相结合的问题对于任何试图在涉及时间扩展行为的情况下提高其性能的分布式学习系统来说都是存在的(Williams,1988)。

例如，当误差-修正学习被用于多层前馈神经元网络时，信任赋值问题就出现了。在这样的网络里，每个隐神经元的运作像每个输出神经元的运作一样，对于网络在一个感兴趣的学习任务上正确的整体运作都是重要的。也就是说，为了解决所规定的任务，网络必须通过误差-修正学习的规范给它的神经元赋予一定的行为方式。在这种背景下，考虑图2-1a描述的情形。由于输出神经元 k 对外界是可见的，就可能给这个神经元提供一个期望响应。就输出神经元而言，根据误差-修正学习来调节输出神经元的突触权值是一件轻而易举的事情，正如2.2节所概括的那样。但是当误差-修正学习过程用于调节隐藏神经元的每个突触权值时，我们如何对这些神经元动作的信任或责任赋值呢？对于这个基本问题的回答需要更详尽的考虑；它在第4章给出，那里描述了设计多层前馈神经网络的算法细节。[62]

2.8 有教师学习

现在让我们把注意力转向学习范例。我们首先讨论有教师学习，也称为有监督学习。图2-6说明这种学习方式的方框图。从概念上讲，我们可以认为教师具有对周围环境的知识(这种类型的知识的形式就是一系列的输入-输出事例)。然而感兴趣的神经网络对这种环境一无所知。现在我们假设教师和神经网络同时要对从周围环境中抽取出来的训练向量(即例子)作出判断，教师可以根据自身掌握的一些知识为神经网络提供对训练样本的期望响应。期望响应一般都代表着神经网络完成的最优动作。神经网络的参数可以在训练向量和误差信号的综合影响下进行调整。*误差信号*可以定义为神经网络实际响应与预期响应之差。这种调整可以逐步而又反复地进行，其最终目的就是要让神经网络模拟教师；在某种统计的意义下，可以认为这种模拟是最优的。利用这种手段，教师对环境掌握的知识就可以由训练最大限度地传授给神经网络。当条件成熟的时候，就可以将教师排除在外，让神经网络完全自主地应对环境。

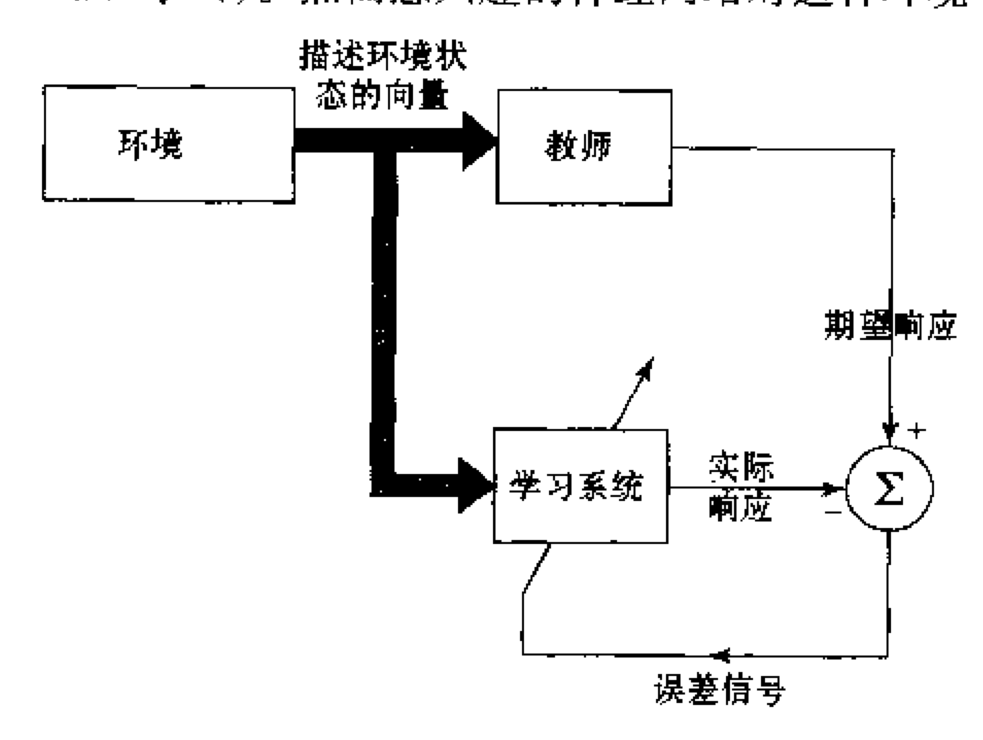


图2-6 有教师学习方框图

我们刚刚描述的有监督学习就是前面2.2节讨论的误差-修正学习方法。它是一种闭环反馈系统，但未知的环境不包含在循环中。我们可以采用训练样本的均方误差或平方误差和作为性能测试手段，它可以定义为系统的一个带自由参数的函数。该函数可以看作一个多维*误差-性能曲面*，或者简称*误差曲面*，其中自由参数作为坐标轴。实际误差曲面是所有可能的输出输入的平均。任何一个在教师监督下的系统给定操作都表示误差面上的一个点。该系统要随时间提高性能，就必须向教师学习，操作点必须要向着误差曲面的最小点逐渐下降，[63] 误差极小点可能是局部最小，也可能是全部点中的最小。有指导学习系统能够处理这些有用

信息，它可以根据系统当前的行为计算出误差曲面的梯度。误差曲面上任何一点的梯度指的是指向最速下降方向的向量。实际上，在向例子进行有监督学习的情况下，系统可以采用梯度向量瞬时估计，这时假如将例子的标号约定为访问的时间。采取这种估计一般会导致在误差曲面上操作点的运动轨迹经常以"随机漫游"的形式出现。然而，如果我们能给定一个设计好的算法来使代价函数最小，而且有足够的输入/输出的数据集和充裕的训练时间，那么有指导学习系统往往可以较好地完成诸如模式分类、函数逼近之类的任务。

2.9 无教师学习

在有监督学习系统中，学习过程是在教师的监督下进行的。然而，在无教师学习范例中，正如它的名字暗示的那样没有教师监视学习过程。也就是说，神经网络没有任何带标号的例子可以学习。第二种学习范例(无监督学习)又分为两类：增强式学习/神经动态规划和无监督学习。

1. 增强式学习/神经动态规划

在增强式学习(reinforcement learning)[8]中，输入输出映射的学习是通过与环境的不断交互来完成的，目的是使一个标量性能指标达到最小。图2-7显示的是增强式学习的方框图。这种学习系统建立在一个评价的基础上，评价将从周围环境中接收到的原始增强信号转换成一种称为启迪增强信号的高质量的增强信号，两者都是标量输入(Barto et al.,1983)。设计该系统的目的是为了适应延迟增强情况下的学习，即意味着系统观察从环境接收的一个时序刺激(即状态向量)，它们最终产生启发式的增强信号。学习的目标是将cost-to-go函数最小化，cost-to-go函数定义为采取一系列步骤的动作代价的累积期望值，而不是简单的直接代价。可以证明：在时间序列上早期采取的动作事实上是整个系统最好的决定。学习机的功能(它构成了系统的第二个组件)就是用来发现这些动作并将它们向环境反馈。

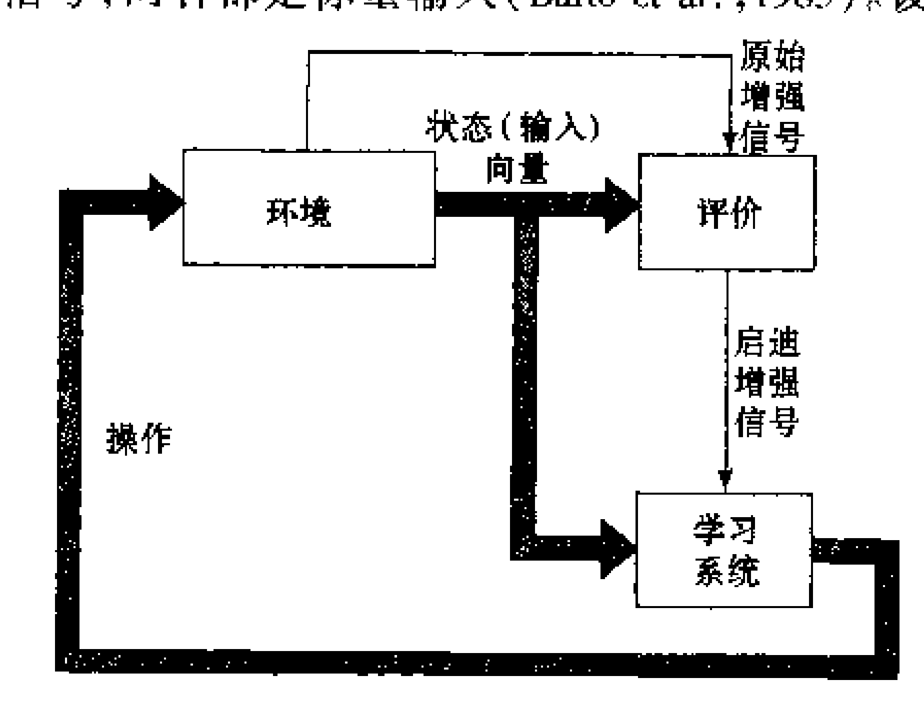


图2-7 增强式学习方框图

64

延迟增强式学习系统很难在实际上运用，基本原因有二：

- 在学习过程中的每个步骤，没有教师提供一个期望的响应。
- 延迟会导致原始增强信号，这意味着学习机必须解决时间信任赋值问题。也就是说，对将导致最终结果的时间序列步中的每一个动作，学习机必须各自独立地对信任和责任赋值，而原始增强可能仅评价最终结果。

尽管存在这些困难，延迟增强学习还是非常有吸引力的。它提供系统与周围环境交互的基础，因此可以仅仅在这种与环境交互获得经验结果的基础上，发展学习完成指定任务的能力。

增强式学习和Bellman(1957)在最优控制理论背景下提出的动态规划密切相关。动态规划提供作出系列决策的数学形式。将增强式学习放在动态规划的框架中，主题就更加丰富，

这一点在 Bertsekas and Tsitsiklis(1996)中作了表述。动态规划的介绍以及它与增强式学习的关系将在第 12 章讨论。

2. 无监督学习

如图 2-8 所示，在无监督或自组织学习系统中，没有外部的教师或者评价来监督学习的过程。提供独立于任务的表示性质的度量，要求网络学习该度量而且自由参数将根据这个度量来逐步优化网络。一旦神经网络能够与输入数据的统计性特征相一致，那么它将发展形成用于输入数据编码特征的内部表示的能力，从而自动创造新的类别(Becker,1991)。

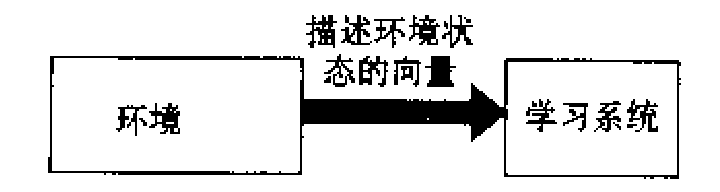


图 2-8 无监督学习方框图

为了完成无监督学习，我们可以使用竞争性学习规则。例如，神经网络可能包括两层：输入层和竞争层。输入层接受有用的数据。竞争层由相互竞争(根据一定的学习规则)的神经元组成，它们力图获得响应包含在输入数据中的特征的“机会”。最简单的形式就是神经网络采用“胜者全得”的策略。正如 2.5 节所述，在这种策略中具有最大总输入的神经元赢得竞争而被激活，其他所有的神经元被关掉。

65

在第 8 章到第 11 章将讨论无监督学习的不同算法。

2.10 学习任务

本章前面几节讨论了不同的学习算法和学习范例。在本节中，我们将描述一些基本的学习任务。选定一个特定的学习算法与神经网络需要完成的学习任务密切相关。在这种背景下，我们将根据不同的形式分别比较神经网络的六种不同的学习任务。

模式联想

联想记忆是与大脑相似的依靠联想学习的分布式记忆。自从亚里士多德时代起，联想就被认作是人脑的一个显著特征，而且认知的所有模式都以这种或那种形式使用联想作为基本的行为(Anderson,1995)。

联想有两种形式：自联想与异联想。自联想方式当存储一系列的模式(向量)时神经网络要求不断地将它们呈现给网络。其后将已存模式的部分描述或畸变(噪声)形式呈现给网络，而网络的任务就是检索(回忆)存储的该特定模式。异联想与自联想的不同之处就在于一个任意的输入模式集合与另一个输出模式集合配对。自联想需要使用无监督学习方式，而异联想采用监督学习方式。

设 $\mathbf{x}_k$ 表示在联想记忆中的关键模式(向量)而 $\mathbf{y}_k$ 表示存储模式(向量)。网络完成的模式联想由

$$\mathbf{x}_k \to \mathbf{y}_k, \qquad k = 1,2,\cdots,q \tag{2.18}$$

表示，其中 q 是存储在网络中的模式数。关键模式 $\mathbf{x}_k$ 作为输入，不仅决定存储模式 $\mathbf{y}_k$ 的存储位置，同时也拥有检索该模式的键码。

在自联想记忆模式中：$\mathbf{x}_k = \mathbf{y}_k$，所以输入输出数据的空间维数相同。在异联想记忆模式中：$\mathbf{x}_k \neq \mathbf{y}_k$；因此，第二种情况的输出空间维数可能与输入数据空间维数相同，也可能不同。

66

联想记忆模式的操作一般包括两个阶段：

- 存储阶段，指的是根据式(2.18)对网络进行训练。
- 回忆阶段，网络根据所呈现的有噪声的或畸变的关键模式检索对应的存储模式。

令刺激(输入)$\mathbf{x}$表示关键模式$\mathbf{x}_j$的有噪声或畸变形式。如图2-9所示，这个刺激产生响应(输出)$\mathbf{y}$。作为完整的回忆，我们将发现$\mathbf{y}=\mathbf{y}_j$，其中$\mathbf{y}_j$为由关键模式$\mathbf{x}_j$联想的记忆模式。如果对$\mathbf{x}=\mathbf{x}_j$有$\mathbf{y}\neq\mathbf{y}_j$，就说联想记忆有回忆错误。



图2-9 模式联想输入输出关系图

联想记忆中存储的模式数目q提供网络存储能力的一个直接度量。在设计联想记忆时，问题就是使存储能力q(表示为与构建网络的神经元总数N的百分比)尽量大，并且保持记忆中的大部分模式能正确回忆。

模式识别

人类非常擅长模式识别，通过感官，我们可以从周围的世界接受到数据，并且可以识别出数据源。我们往往是瞬间完成，几乎毫不费力。例如，我们能够识别出任何一张熟悉的脸，即使我们和这个人已经多年未曾谋面。无论电话线路如何差劲，我们还是可以迅速地根据他或者她的声音很快地甄别出你的熟人。仅仅闻一下，就能分辨出一个煮鸡蛋是否变坏。人类是通过学习过程来成功地实现模式识别的，神经网络也是如此。

模式识别被形式地定义为一个过程，由这个过程将接收的模式或信号确定为一些指定类(类别)中的一个类。一个神经网络要实现模式识别需要先经过一个训练的过程，在此过程中网络需要不断地接受一个模式集合以及每个特定模式所属的类别；然后，把一个以前没有见过但属于用于训练网络的同一模式总体的新模式呈现给神经网络。神经网络可以根据从训练数据中提取的信息识别特定模式的类别。神经网络的模式识别本质上是基于统计特性的，各个模式可以表示成为多维判定空间的一些点。判定空间被划分为不同的区域，每个区域对应一个模式类。判定边界由训练过程决定。我们可以根据各个模式类内部以及它们之间固有可变性用统计方式确定边界。

67

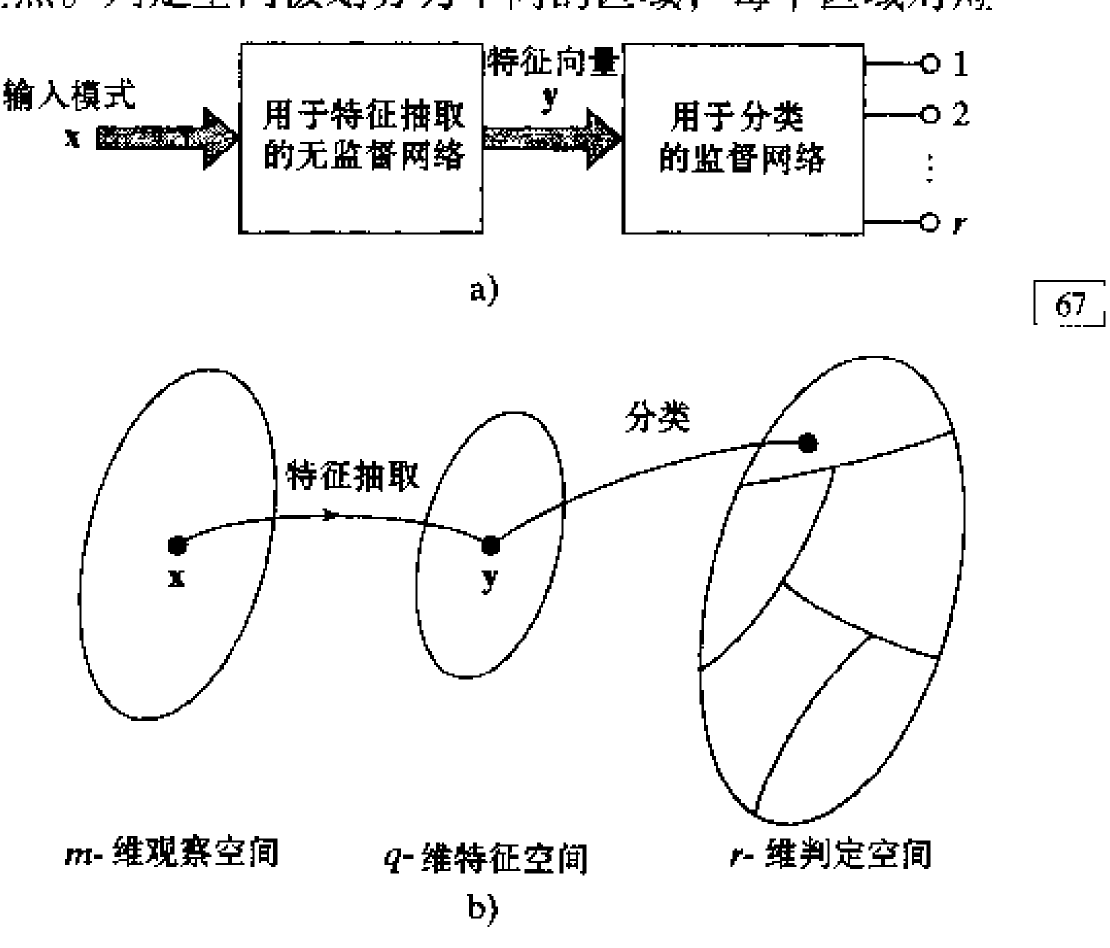


图2-10 模式分类的经典分类方法图解

一般而论，采用神经网络的模式识别机分为如下两种形式：

- 如图2-10a所示，识别机分为两部分，用来作特征抽取的无监督网络和作分类的监督网络。这种方法遵循传统的统计特性模式识别方法(Duda and Hart, 1973; Fukunaga, 1990)。用概念术语来表示，一个模式是一个m维的可观测的数据，即m维观测

(数据)空间集中的一个点 $\mathbf{x}$。如图 2-10b 所示，特征抽取描述为一个变换，它将点 $\mathbf{x}$ 映射成一个 q 维特征空间相对应的中间点 $\mathbf{y}(q < m)$。这种变换可看作是维数缩减(即，数据压缩)，这种做法主要是基于简化分类任务的考虑。分类本身可描述为一个变换，它将中间点 $\mathbf{y}$ 映射为 r 维判定空间上的一个类，其中 r 是要区分的类别数。

- 识别机设计成一个采用监督学习算法的多层前馈网络。在这第二个方法中，特征抽取由网络隐藏层中的计算单元执行。

实际应用中到底采用两个方法中的哪一个方法，取决于实际应用的着眼点。

函数逼近

第三个学习任务是函数逼近。考虑由函数关系

68

$$\mathbf{d} = \mathbf{f}(\mathbf{x}) \tag{2.19}$$

描述的一个非线性输入输出映射，其中向量 $\mathbf{x}$ 是输入，向量 $\mathbf{d}$ 为输出。向量值函数 $\mathbf{f}(\cdot)$ 假定为未知。为了弥补函数 $\mathbf{f}(\cdot)$ 知识的缺乏，我们假定有如下的训练样例集合：

$$\mathcal{T} = \{(\mathbf{x}_i, \mathbf{d}_i)\}_{i=1}^{N} \tag{2.20}$$

我们的要求是设计一个神经网络来逼近未知函数 $\mathbf{f}(\cdot)$，使由网络实际实现的描述输入-输出映射的函数 $\mathbf{F}(\cdot)$ 在欧几里德距离的意义下与 $\mathbf{f}(\cdot)$ 足够接近，即

$$\|\mathbf{F}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\| < \varepsilon, \text{对于所有的 } \mathbf{x} \tag{2.21}$$

其中 ε 是一个很小的正数。假定训练集样本数目 N 足够大，神经网络也有适当数目的自由参数，那么对于特定的任务逼近误差 ε 应当是足够的小。

在这里，逼近问题其实是一个很完整的监督学习，其中 $\mathbf{x}_i$ 是输入向量，而 $\mathbf{d}_i$ 是期望的响应。我们可以换一个角度思考这种问题，将监督学习看成是一个逼近问题。

神经网络逼近一个未知输入-输出映射的能力可以从两个重要途径利用：

- *系统辨识*。假定式(2.19)描述的是一个未知的无记忆的多输入-多输出(multiple input-multiple output，MIMO)系统的输入输出关系；所谓“无记忆”系统，我们指的是时间不变性的系统。然后我们利用在式(2.20)中的标定的例子集合将神经网络训练为系统的一个模型。假定 $\mathbf{y}_i$，表示神经网络中对输入向量 $\mathbf{x}_i$ 产生的相应输出。正如图 2-11 所描绘，$\mathbf{d}_i$(与 $\mathbf{x}_i$ 相对应)与输出 $\mathbf{y}_i$，之间产生一个误差信号 $\mathbf{e}_i$，这个误差信号接着用来调节网络的自由参数，最终使未知系统的输出和神经网络输出在整个训练集上的平方差在统计意义上达到最小。

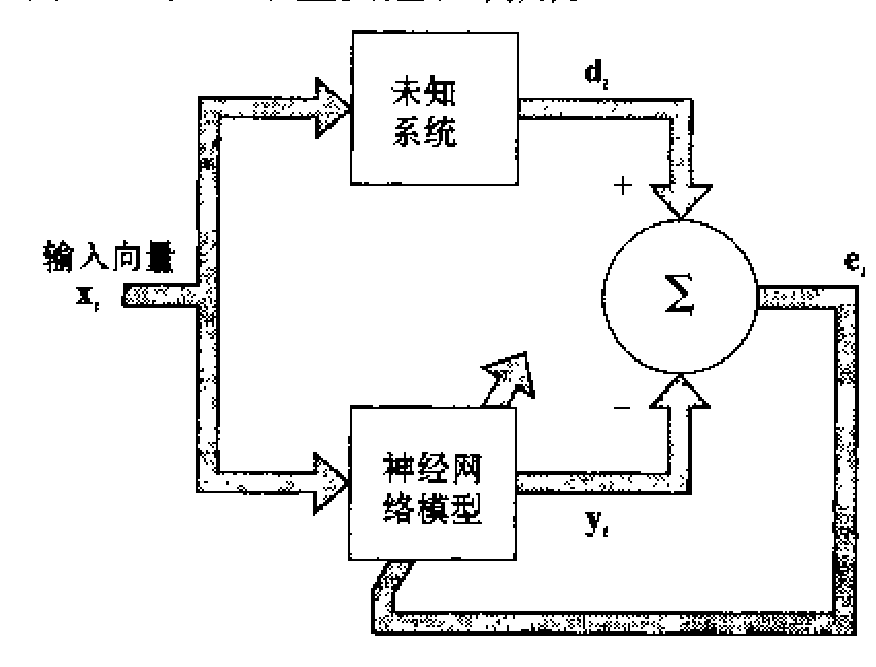


图 2-11 系统识别方框图

- *逆系统*。下一步假定我们给定一个已知无记忆 MIMO 系统，其中输入输出关系如式(2.19)所示。在这种情况下的要求是如何构造一个*逆系统*，针对向量 $\mathbf{d}$ 产生系统向量 $\mathbf{x}$。逆系统可以由

69

$$\mathbf{x} = \mathbf{f}^{-1}(\mathbf{d}) \tag{2.22}$$

描述，其中向量值函数 $\mathbf{f}^{-1}(\cdot)$ 表示 $\mathbf{f}(\cdot)$ 的反函数。注意，$\mathbf{f}^{-1}(\cdot)$ 不是 $\mathbf{f}(\cdot)$ 的倒数，上标 -1 仅仅是反函数的标志而已。在实际遇到的很多问题中，向量值函数 $\mathbf{f}(\cdot)$ 过于复杂，从而限制了求出反函数 $\mathbf{f}^{-1}(\cdot)$ 的直接公式。给定如式(2.20)的一些样例集，我们可以通过采取图 2-12 所示的过程构造一个神经网络来逼近函数 $\mathbf{f}^{-1}(\cdot)$。在这里描述的情况中，$\mathbf{x}_i$ 和 $\mathbf{d}_i$ 的作用交换了位置：向量 $\mathbf{d}_i$ 作为输入，向量 $\mathbf{x}_i$ 作为期望的响应。假定向量 $\mathbf{e}_i$ 表示 $\mathbf{x}_i$ 与神经网络针对 $\mathbf{d}_i$ 的实际输出 $\mathbf{y}_i$ 之间的误差。与系统辨识问题类似，利用误差信号向量来调节网络的自由参数，最终使未知逆系统的输出和神经网络输出在整个训练样例集上的平方差在统计意义上达到最小。

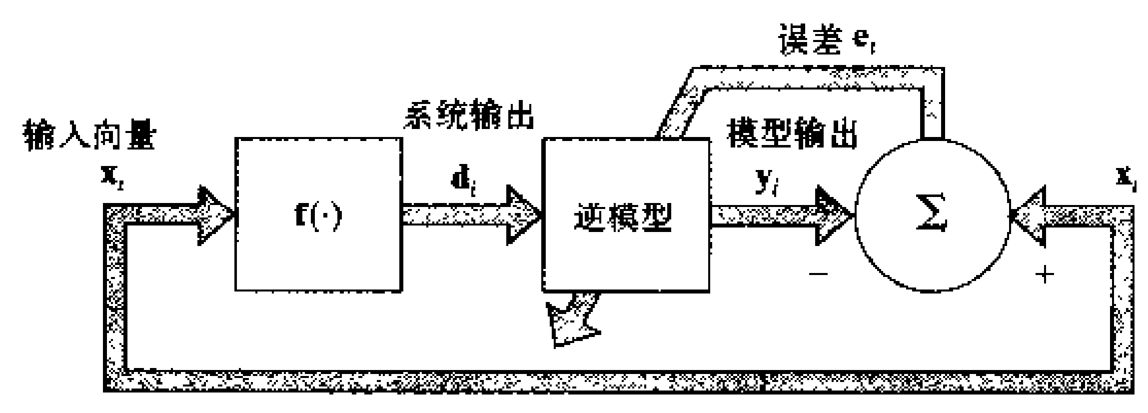


图 2-12 逆模式系统方框图

控制

神经网络可以完成的另外一个学习任务是对设备进行控制操作。所谓“设备”指的是一个过程或者是可以在被控状态下维持运转的系统的一个关键部分。学习和控制相关其实不是一件什么值得大惊小怪的事情，毕竟我们人脑就是一个计算机(即信息处理器)，作为整个系统的输出是实际的动作。在控制的这种意义下，人脑就是一个活生生的例子，它证明可以建立一个广义控制器，充分利用并行分布式硬件，能够并行控制成千上万的致动器(如肌肉神经纤维)，能够处理非线性性和噪声，并且可以在长期计划水平上进行优化(Werbos，1992)。

考虑如图 2-13 所示的反馈控制系统。该系统涉及利用被控设备的单元反馈，即设备的输出直接反馈给输入[9]。因此设备的输出 $\mathbf{y}$ 减去从外部信息源提供的参考信号 $\mathbf{d}$。这样最终产生误差信号 $\mathbf{e}$ 并将之应用到神经控制器以便调节它的自由参数。控制器的主要功能就是为设备提供相应的输入，从而使它的输出 $\mathbf{y}$ 跟踪参考信号 $\mathbf{d}$。换句话说，就是控制器不得不对设备的输入输出行为进行转换。 70

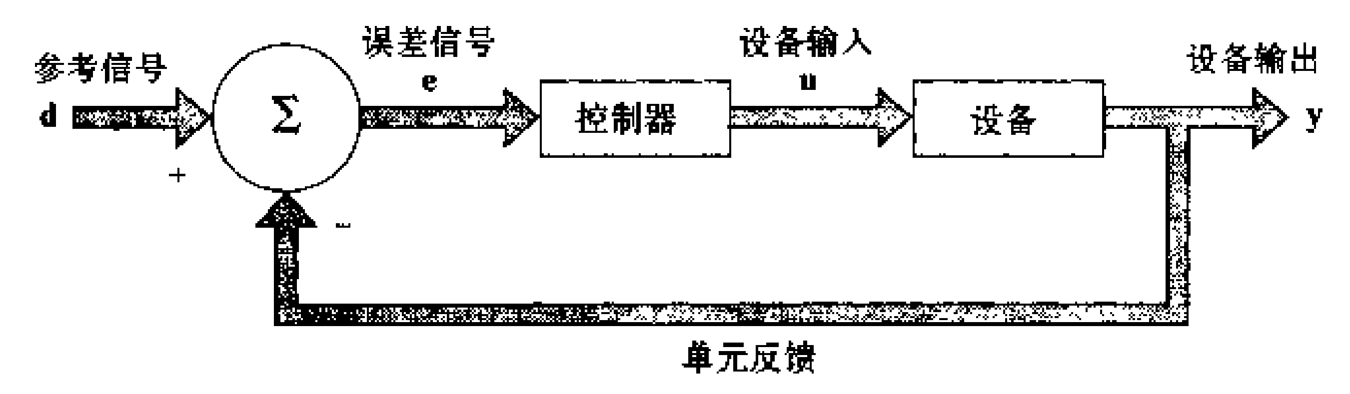


图 2-13 反馈控制系统方框图

我们注意到在图 2-13 中误差信号 $\mathbf{e}$ 在到达设备之前先通过神经控制器。结果，根据误差-修正学习算法为了实现对设备自由参数的调节，我们必须知道 Jacobi 矩阵

$$\mathbf{J} = \left\{\frac{\partial y_k}{\partial u_j}\right\} \tag{2.23}$$

其中 y_k 中是设备输出 $\mathbf{y}$ 的一个元件，而 u_j 是设备输入 $\mathbf{u}$ 的一个元件。不幸的是偏导数$\partial y_k/\partial u_j$对于不同的$k$，$j$依赖于设备的运行点，因而是未知的。我们可以采用下面两种方法之一来近似计算该偏导数：

- *间接学习*。利用设备的实际输入－输出测量值，首先构造神经网络模型产生一个它的复制品。接着利用这个复制品提供 Jacobi 矩阵 $\mathbf{J}$ 的一个估计值。随之把构成 Jacobi 矩阵 $\mathbf{J}$ 的偏导数用于误差－修正学习算法，以便计算对神经控制器的自由参数的调节(Nguyen and Widrow,1989;Suykens et al.,1996;Widrow and Walach,1996)。
- *直接学习*。偏导数$\partial y_k/\partial u_j$ 的符号通常是知道的而且在设备的动态区域内一般是不变的。这意味着我们可以通过各自的符号来逼近这些偏导数。它们的绝对值由神经控制器的自由参数的一种分布式表示给出(Saerens and Soquet,1991;Schiffman and Geffers,1993)。因此，神经控制器能够直接从设备学习如何调节它的自由参数。

滤波

滤波器这个术语一般指的是一种设备或算法，利用它能从一个带有噪声的数据集中抽取一定数量的符合要求的信息。噪声可能是由不同来源引起的。例如，可能是采用带噪声的传感器测量数据，也可能表示承载信息的信号通过通信信道传输时受到损坏。另外一个例子是一个有用的信号元件受到从它周围环境接收的干扰信号的损害。我们可以使用滤波器来实现三个基本的信息处理任务：

1. *滤波*。这个任务指的是在离散的时间 n 用直到 n 且包括 n 在内的测量数据抽取一定量有价值的信息。

2. *平滑处理*。第二个任务不同于滤波处理之处在于在时间 n 内一定量有价值的信息不可得到，而且在时间 n 之后测量到的数据可以用来得到这个信息。这意味着在平滑处理过程中，产生输出结果有延迟。因为在平滑处理过程中，我们不仅能够利用直到时间 n 的数据，而且可以利用在 n 之后的数据，从统计学意义上讲，我们期望平滑过程应当比单纯的过滤更加精确。

3. *预测*。这个任务是指信息处理过程的预测方面。它的目的是通过测量到 n(含 n)时刻的数据，导出一定量有价值的信息，这段信息可能与将来 $n+n_0$ 时刻的数据相似，其中 $n_0>0$。

滤波问题是大家都很熟悉的“鸡尾酒会问题”。[10] 在鸡尾酒会这样一个嘈杂的环境里面，房间里还有其他的干扰性谈话，说话者的声音信号往往埋没于与之几乎差不多的噪声环境中。但无论怎样吵，人们都有一个非常了不起的能力：全神贯注听清与之对话者的谈话。在解决鸡尾酒会问题时，可想而知的是，肯定采取了某种形式的预处理分析手段(Velmans,1995)。在(人工)神经网络环境中，出现一个相似的滤波问题，即盲信号的分离问题(Comon,1994;Bell and Sejnowski,1995;Amari et al.,1996)。为了将盲信号分离问题形式化，我们假定未知源信号集合$\{s_i(n)\}_{i=1}^{m}$彼此之间相互独立。这些信号由未知传感器的线性混合，产生$m\times 1$观察向量(参看图 2-14)

$$\mathbf{x}(n)=\mathbf{A}\mathbf{u}(n) \qquad (2.24)$$

图 2-14 盲源分离方框图

其中

$$\mathbf{u}(n) = [u_1(n), u_2(n), \cdots, u_m(n)]^T \tag{2.25}$$

$$\mathbf{x}(n) = [x_1(n), x_2(n), \cdots, x_m(n)]^T \tag{2.26}$$

而且 **A** 是一个未知的 $m \times m$ 非奇异混合矩阵。给定观察向量 $\mathbf{x}(n)$，要求在无监督方式下恢复原始信号 $u_1(n), u_2(n), \cdots, u_m(n)$。

现在回到预测问题上来，给定过程在过去时间上均匀分布的一些值，如 $x(n-T)$，$x(n-2T), \cdots, x(n-mT)$，其中 T 式采样周期，m 是预测顺序，要求对过程的当前值 $x(n)$ 作出预测。如图 2-15 所示，既然训练样本是直接从过程本身来抽取的，可以利用监督学习的误差－修正方法来解决预测问题，其中 $x(n)$ 假定为期望的响应。假定 $\hat{x}(n)$ 为神经网络在时间 n 产生的预测值，那么误差信号 $e(n)$ 可以定义为 $\hat{x}(n)$ 与 $x(n)$ 的差值，$e(n)$ 用来调节神经网络的自由参数。基于此，预测可视为某种形式上的模型构建，在统计意义下，这种预测误差越小，网络作为产生数据的内在物理过程的模型性能就越好。如果这一过程是非线性的，那么使用神经网络就为解决预测问题提供了一个强有力的解决方案，因为非线性处理单元可以嵌入它的构造中。但是使用非线性处理单元惟一可能的例外是网络的输出单元。如果时间数列 $\{x(n)\}$ 的动态区域是未知的，最合理的选择是使用线性输出单元。

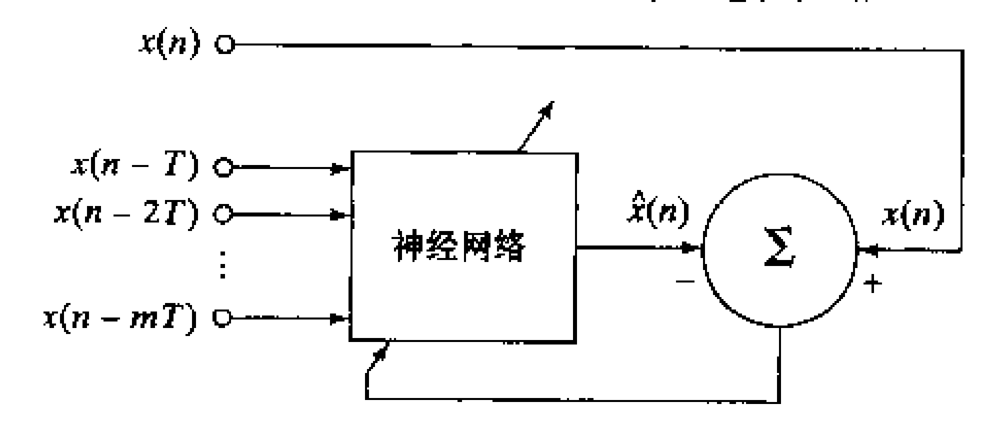


图 2-15 非线性预测方框图

波束形成

波束形成是滤波的空间形式，利用它区分目标信号和背景噪声的空间性质。用于波束形成的设备称为*波束形成器*。

波束形成的任务适合利用神经网络，因为从人类听觉反应的心理声学的研究（Bregman, 1990）和蝙蝠回声定位听觉系统皮质层的特征映射研究（Suga, 1990a; Simmons and Sailant, 1992）中，我们有了相关的线索。蝙蝠的回声定位由发送短时频率调制（frequency-modulated, FM）声纳信号了解周围环境，然后利用它的听觉系统（包括一对耳朵）集中注意于它的猎物（如飞行的昆虫）。蝙蝠的耳朵提供某种形式的空间滤波（准确地说为空间干扰测量术），听觉系统利用它产生*注意的选择性*（attentional selectivity）。

波束形成通常用于雷达和声纳系统，它们的基本任务是在接收器噪声和干扰信号（如人为干扰）出现的情况下探测和跟踪感兴趣的目标。两个因素使这个任务复杂化。

- 目标信号源自未知的方向。
- 干扰信号无可用的先验信息。

73

处理这种情况的一种方法是使用*广义旁瓣消除器*（generalized sidelobe canceller, GSLC），图 2-16 显示的是它的方框图。这个系统由以下组件组成（Griffiths and Jim, 1982; Van Veen, 1992; Haykin, 1996）：

- 一个*天线元阵列*，它提供对空间中离散点上的被观察的信号取样的手段。
- 一个*线性组合器*，它是由固定的权重集合 $\{w_i\}_{i=1}^m$ 定义的，其输出就是期望的响应。这个线性组合器的作用就像一个"空间滤波器"，它由一个辐射模式刻画（例如，一个

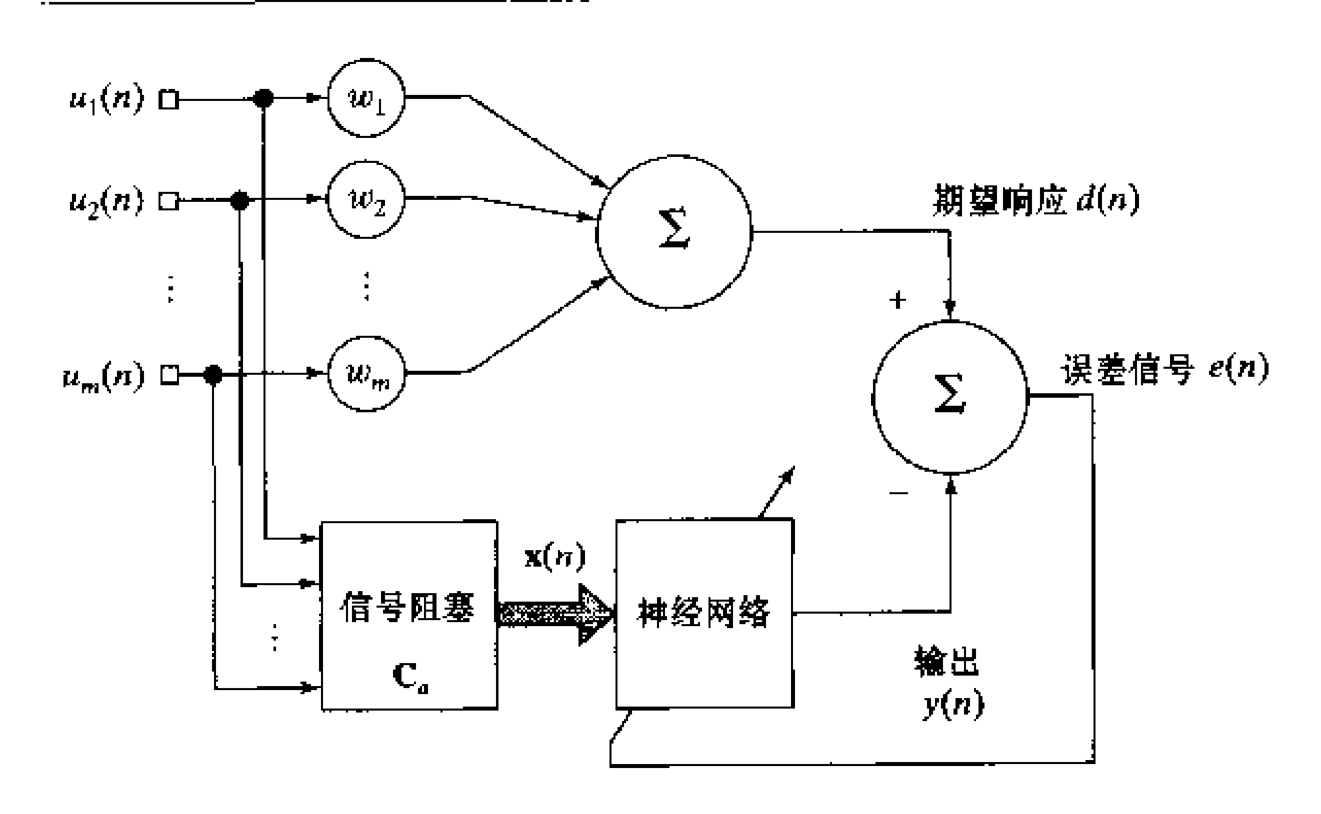


图 2-16 广义旁瓣消除器方框图

天线输出振幅与输入信号入射角的极坐标图)。辐射模式的主瓣指向规定的方向。因此 GSLC 受它约束而产生一个无畸变的响应。线性组合器的输出记为 $d(n)$，它对波束形成器提供期望的响应。

- 一个信号阻塞矩阵 $\mathbf{C}_a$，它的功能是删除干扰，这种干扰是通过代表线性组合器的空间滤波器辐射模式的旁瓣泄漏的。
- 一个具有可调参数的神经网络，它被设计成能适应干扰信号的统计变化。

神经网络的自由参数的调节是由一个在误差信号 $e(n)$ 上操作的纠错学习算法完成的，$e(n)$由线性组合器的输出 $d(n)$和神经网络的实际输出 $y(n)$之间的差确定。从而 GSLC 在线性组合器的监督下操作，线性组合器担当着"教师"的角色。作为普通的监督学习时，注意线性组合器是在神经网络的反馈环之外的。一个使用神经网络来学习的波束形成器称为神经波束形成器(neural beamformer)或者神经 - 波束形成器(neuro-beamformer)。这类学习机可归入注意性神经计算机(attentional neurocomputers)的范围(Hecht-Nielsen,1990)。

这里讨论的 6 个学习任务的多样性是神经网络作为信息处理系统通用性的证明。从基本意义上说，这些学习任务都是从映射的样例中(可能有噪声)学习映射的问题。如果没有强迫接受先验知识，可能的解映射并不惟一，从这个意义上来说，每个任务事实上都是不适定的。使这些解适定的一个方法是使用第 5 章描述的正则化理论。

74

2.11 记忆

关于学习任务的讨论，特别是模式联想的任务，使我们很自然地考虑记忆的问题。在神经生物学的语义环境中，记忆是指由生物和它的环境之间相互作用而诱导出的相对持久的神经改变(Teyler,1986)。没有这种变化就没有记忆。而且，要想这种记忆有用，它必须对神经系统是可存取的，这样才可以去影响未来的行为。然而，一个活跃模式必须首先通过学习过程被存储在记忆里，记忆和学习错综复杂地联系着。当一个特定的活跃模式被学习后，它就存放在脑中某个地方，在需要时就会回忆起来。记忆可以分为"短期"和"长期"记忆，取决于保持的时间(Arbib,1989)。短期记忆指代表环境的"当前"状态的知识的编制。以短期记忆存储的知识和"新"的状态之间的任何差异，都会用来更新短期记忆。另一方面，长期记忆指长

时间或永远存储的知识。

在这一节中，我们学习有如下特征的联想记忆：

- 记忆是分布式的。
- 联想记忆的刺激(关键)模式和响应(存储)模式由数据向量组成。
- 通过设置大量神经元的神经活动的空间模式，在记忆里存储信息。
- 刺激包含的信息不仅决定它在记忆中的存储位置而且决定它的检索地址。
- 虽然神经元不代表可靠的和低噪音的计算元，但是记忆表现出对扩散类型的噪音和破坏的高度抑制。
- 存储在记忆中的单个模式之间应该有相互作用。(否则记忆将会变得非常大，因为它要去适应大量彼此完全隔离的模式的存储。)这就是对于记忆在回忆过程中产生误差的独特的可能性。

在分布式记忆中，基本的问题是许多不同神经元的同时或接近同时的行动，这是外部或内部刺激的结果。神经活动在记忆内构成的空间模式包含关于刺激的信息。因此，我们说记忆去执行一个分布式映射，它把一个输入空间的活跃模式转换为另一个输出空间活跃模式。我们可以考虑一个理想化的由两层神经元组成的神经网络，来解释分布式记忆映射的一些重要特性。可以认为图2-17是神经系统组件模型的网络的图解(Cooper, 1973; Scofield & Cooper, 1985)。在输入层的每个神经元都和输出层的每个神经元相联结。实际上突触之间的连接是复杂的和有冗余的。在图2-17a的模型里，一个单独的理想连接被用来表示所有突触联系之间的整合作用，这些突触联系存在于输入层的神经树突和输出层的神经轴突分支之间。输入层一个神经元的活动水平会影响到输出层每个神经元的活动水平。

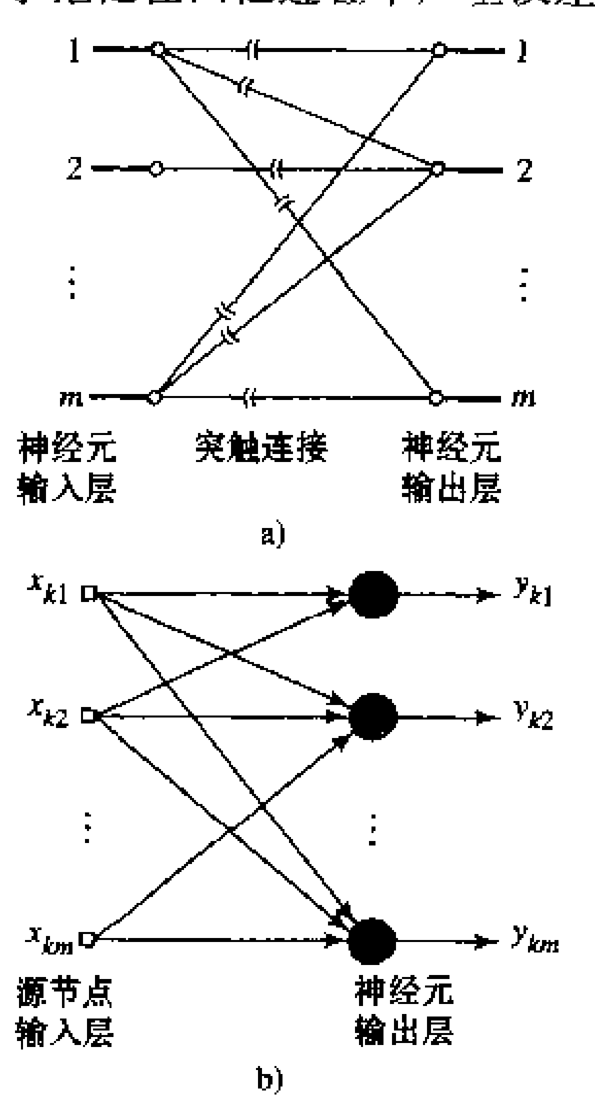


图2-17 联想记忆模型
a)神经系统的联想记忆模型组件
b)使用人工神经元的联想记忆模型

75

图2-17b描绘的是相应的人工神经网络的情况。图中有一个源节点的输入层和一个作为计算节点的神经元输出层。在这种情况下，网络的突触权重被作为神经元的整体部分包括在输出层。网络的两层之间的连接链是简单连线。

在以下的数学分析中，假定图2-17a和2-17b的神经网络是线性的。这一假设的内涵是每一个神经元都像一个线性组合器一样运作，如图2-18的信号流图所示。为了进行分析，设想一个活动模式 $\mathbf{x}_k$ 发生在网络的输入层，另一个活动模式 $\mathbf{y}_k$ 同时发生在输出层。这里我们要考虑的问题是从模式 $\mathbf{x}_k$ 和模式 $\mathbf{y}_k$ 之间的联想中学习。模式 $\mathbf{x}_k$ 和 $\mathbf{y}_k$ 用向量表示，它们的扩展形式记为：

$$\mathbf{x}_k = [x_{k1}, x_{k2}, \cdots, x_{km}]^T$$

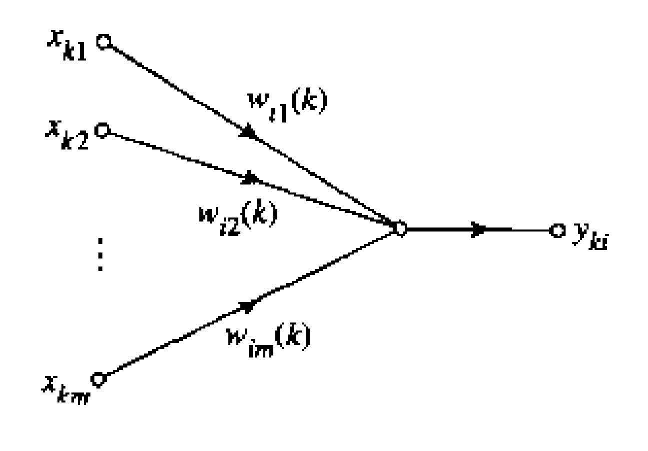


图2-18 线性神经元 i 的信号流图模型

76

和

$$\mathbf{y}_k = [y_{k1}, y_{k2}, \cdots, y_{km}]^T$$

为了表达的方便，我们假定输入空间维数(例如向量 $\mathbf{x}_k$ 的维数)和输出空间维数(例如向量 $\mathbf{y}_k$ 的维数)是相同的，都等于 m。从此我们称 m 为网络维数或者简称维数。注意 m 等于输入层的源节点数目或者输出层的神经元数目。对于一个有着大量神经元的神经网络，这是典型的情况，维数 m 可能是很大的。

$\mathbf{x}_k$ 和 $\mathbf{y}_k$ 的元素可假定是正值或负值。这是人工神经网络中一个合理的假定。当考虑一个真实的激活水平(例如一个神经元的激发率)和一个非零的自发激活水平之间差异作为相关的生理学变量时，它也可能发生在一个神经系统中。

图2-17假定为线性的网络，关键向量 $\mathbf{x}_k$ 和已记忆的向量 $\mathbf{y}_k$ 之间的联想可以用矩阵的形式来表示如下：

$$\mathbf{y}_k = \mathbf{W}(k)\mathbf{x}_k, \qquad k = 1,2,\cdots, q \tag{2.27}$$

这里 $\mathbf{W}(k)$是权值矩阵，单独由输入-输出对$(\mathbf{x}_k, \mathbf{y}_k)$确定。

为了显示权值矩阵 $\mathbf{W}(k)$的详细的描述，考虑图2-18，它显示输出层里一个神经元 i 的具体排列。由于输入层刺激用于对关键模式 $\mathbf{x}_k$ 的元素的组合动作，神经元 i 的输出 y_{ki} 由下式给出：

$$y_{ki} = \sum_{j=1}^{m} w_{ij}(k)x_{kj}, \qquad i = 1,2,\cdots,m \tag{2.28}$$

其中 $w_{ij}(k)$，$j = 1, 2, \cdots, m$ 是神经元i对应于第k对联想模式的突触权重。使用矩阵记号，我们可以用等价的形式

[77]

$$y_{ki} = [w_{i1}(k), w_{i2}(k), \cdots, w_{im}(k)]\begin{bmatrix} x_{k1} \\ x_{k2} \\ \vdots \\ x_{km} \end{bmatrix}, \qquad i = 1,2,\cdots,m \tag{2.29}$$

表达 y_{ki}。式(2.29)右边的列向量被识别为关键向量 $\mathbf{x}_k$，通过把式(2.29)代入 $m \times 1$ 的存储向量 $\mathbf{y}_k$ 的定义，得到

$$\begin{bmatrix} y_{k1} \\ y_{k2} \\ \vdots \\ y_{km} \end{bmatrix} = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \cdots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \cdots & w_{2m}(k) \\ \vdots & \vdots & \vdots & \vdots \\ w_{m1}(k) & w_{m2}(k) & \cdots & w_{mm}(k) \end{bmatrix}\begin{bmatrix} x_{k1} \\ x_{k2} \\ \vdots \\ x_{km} \end{bmatrix} \tag{2.30}$$

式(2.30)是式(2.27)中描述的矩阵变换或映射的展开形式，特别是，$m \times m$ 矩阵 $\mathbf{W}(k)$可以定义为

$$\mathbf{W}(k) = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \cdots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \cdots & w_{2m}(k) \\ \vdots & \vdots & \vdots & \vdots \\ w_{m1}(k) & w_{m2}(k) & \cdots & w_{mm}(k) \end{bmatrix} \tag{2.31}$$

单独地表示 q 对联想模式 $\mathbf{x}_k \to \mathbf{y}_k$，$k = 1, 2, \cdots, q$，生成每个矩阵相应的值，即 $\mathbf{W}(1)$，$\mathbf{W}(2)$，$\cdots$，$\mathbf{W}(q)$。假如这个联想模式用权值矩阵 $\mathbf{W}(k)$代替，我们就可以定义一个

$m \times m$ 记忆矩阵，用来描述整个联想模式集合的权值矩阵的总和，表示如下：

$$\mathbf{M} = \sum_{k=1}^{q} \mathbf{W}(k) \tag{2.32}$$

记忆矩阵 **M** 定义联想记忆的输入和输出层之间的全部连接。事实上，矩阵 **M** 代表记忆表述 q 个输入－输出模式获得到的总经验。用另一种方式表示就是，记忆矩阵 **M** 包含有每个出现在记忆中的活动模式的输入输出对。

式(2.32)中给出的关于记忆矩阵的定义用递归的形式可以重新表示为

$$\mathbf{M}_k = \mathbf{M}_{k-1} + \mathbf{W}(k), \qquad k = 1,2,\cdots,q \tag{2.33}$$

这里 $\mathbf{M}_0$ 的初值是 0(也就是说，记忆中的所有突触权值都被初始化为 0)，最终的值 $\mathbf{M}_q$ 和式(2.32)中定义的 **M** 的值完全相等。根据递归公式(2.33)可知，项 $\mathbf{M}_{k-1}$ 是从$(k-1)$个联想模式得出的记忆矩阵的旧值，$\mathbf{M}_k$ 是按照第 k 个联想模式产生的增量 $\mathbf{W}(k)$更新后的值。然而，要注意的是，如果把 $\mathbf{W}(k)$加到 $\mathbf{M}_{k-1}$上，增量 $\mathbf{W}(k)$的值就失去了在组成 $\mathbf{M}_k$ 时的惟一性。虽然考虑不同联想的突触混合，但有关刺激的信息可能并未丢失，就像最后结果显示的那样。78 还要注意的是，当存储的模式数量 q 增大时，记忆中新模式的影响总的来说在逐渐减小。

相关矩阵记忆

假设图 2-17b 的联想记忆通过由 $\mathbf{x}_k \to \mathbf{y}_k$ 描述的关键模式和记忆模式的联想学习了记忆矩阵 **M** 并已经记住了，这里 $k = 1, 2, \cdots, q$。我们可以假定 $\hat{\mathbf{M}}$，代表记忆矩阵 **M** 根据这些模式得出的估计值(Anderson, 1972, 1983; Cooper, 1973)，表示如下：

$$\hat{\mathbf{M}} = \sum_{k=1}^{q} \mathbf{y}_k \mathbf{x}_k^T \tag{2.34}$$

项 $\mathbf{y}_k \mathbf{x}_k^T$ 代表关键模式 $\mathbf{x}_k$ 和记忆模式 $\mathbf{y}_k$ 的*外积*。这个外积是权值矩阵 $\mathbf{W}(k)$的估计值，*权值矩阵*把模式 $\mathbf{y}_k$ 映射到输出模式 $\mathbf{x}_k$ 上。既然模式 $\mathbf{x}_k$ 和 $\mathbf{y}_k$ 都被假设为 $m \times 1$ 向量，所以它们的输出乘积 $\mathbf{y}_k \mathbf{x}_k^T$，也就是估计值 $\hat{\mathbf{M}}$ 就是一个 m 行 m 列的矩阵。这个维数正好和等式(2.32)中定义的记忆矩阵 **M** 相一致。估计值 $\hat{\mathbf{M}}$ 总和的形式与式(2.32)中定义的记忆矩阵有着直接的联系。

外积 $\mathbf{y}_k \mathbf{x}_k^T$ 的典型形式可以表示为 $y_{ki} x_{kj}$，这里 x_{kj} 是输入层中源节点 j 的输出，y_{ki} 是输出层中神经元 i 的输出。在第 k 个联想的突触权植 $w_{ij}(k)$中，源节点 j 代表一个前突触节点，输出层中的神经元 i 代表一个后突触节点。因此，式(2.43)中描述的"局部"学习过程可以看成是 Hebb *学习假设的推广*。考虑到用于构造记忆矩阵 $\hat{\mathbf{M}}$ 的矩阵运算，它也称为*外积规则*。相应地，这样设计的联想记忆称为*相关矩阵记忆*。这种或那种形式的相关，确实是人类神经系统中学习、联想、模式识别和记忆回想的基础(Eggermont, 1990)。

式(2.34)可以重新写成等价的形式为：

$$\hat{\mathbf{M}} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_q] \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_q^T \end{bmatrix} = \mathbf{Y}\mathbf{X}^T \tag{2.35}$$

这里

79

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_q] \tag{2.36}$$

$$\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_q] \tag{2.37}$$

矩阵 $\mathbf{X}$ 是一个由学习过程中使用的所有关键模式组成的 m 行 q 列矩阵。矩阵 $\mathbf{Y}$ 是由相应的记忆模式组成的 m 行 q 列矩阵；称作被记忆矩阵。

式(2.35)可以用递归的形式表示成

$$\hat{\mathbf{M}}_k = \hat{\mathbf{M}}_{k-1} + \mathbf{y}_k\mathbf{x}_k^T, \qquad k = 1,2,\cdots,q \tag{2.38}$$

图2-19表示这个递归的信号流图。根据这个信号流图和递归公式(2.38)，矩阵 $\hat{\mathbf{M}}_{k-1}$ 代表记忆矩阵的旧估计值；矩阵 $\hat{\mathbf{M}}_k$ 代表记忆作用于模式 $\mathbf{x}_k$ 和 $\mathbf{y}_k$ 所得的新联想的改变值。比较式(2.38)和(2.33)中的递归，我们可以看出，外积 $\mathbf{y}_k\mathbf{x}_k^T$ 代表权值矩阵 $\mathbf{W}(k)$ 相应于第 k 个关键模式 $\mathbf{x}_k$ 和记忆模式 $\mathbf{y}_k$ 联想的估计值。

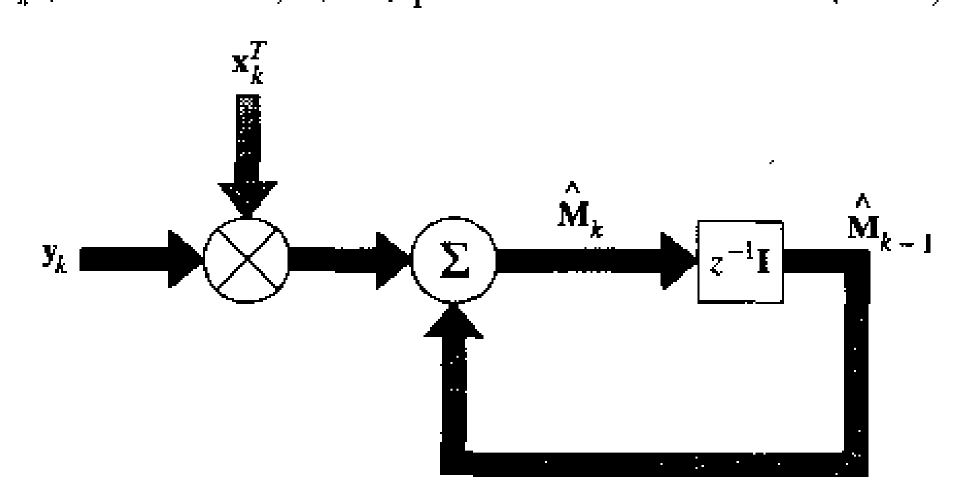


图2-19 等式(2.38)的信号流图表示

回忆

在使用一个联想记忆的过程中提出的基础问题是：保存在记忆中模式的地址和回忆。为了解释这个问题的一个方面，我们让 $\hat{\mathbf{M}}$ 表示一个联想记忆的记忆矩阵，通过与式(2.34)相一致的 q 个联想模式，已经完成了对这个矩阵的学习。随机选取一个关键模式 $\mathbf{x}_j$ 作为记忆的刺激产生响应

$$\mathbf{y} = \hat{\mathbf{M}}\mathbf{x}_j \tag{2.39}$$

将式(2.34)代入式(2.39)，得到

$$\mathbf{y} = \sum_{k=1}^{m}\mathbf{y}_k\mathbf{x}_k^T\mathbf{x}_j = \sum_{k=1}^{m}(\mathbf{x}_k^T\mathbf{x}_j)\mathbf{y}_k \tag{2.40}$$

这里，看到第二行中的 $\mathbf{x}_k^T\mathbf{x}_j$ 是一个标量，它的值等于关键向量 $\mathbf{x}_k$ 和 $\mathbf{x}_j$ 的内积。我们可以将式(2.40)重写成

80

$$\mathbf{y} = (\mathbf{x}_j^T\mathbf{x}_j)\mathbf{y}_j + \sum_{\substack{k=1 \\ k\neq j}}^{m}(\mathbf{x}_k^T\mathbf{x}_j)\mathbf{y}_k \tag{2.41}$$

设关键模式 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_q$ 均被规格化为具有单位能量，即

$$E_k = \sum_{l=1}^{m}x_{kl}^2 = \mathbf{x}_k^T\mathbf{x}_k = 1, \qquad k = 1,2,\cdots,q \tag{2.42}$$

相应地，可以将记忆对刺激(关键模式)$\mathbf{x}_j$ 的响应简化为

$$\mathbf{y} = \mathbf{y}_j + \mathbf{v}_j \tag{2.43}$$

其中

$$\mathbf{v}_j = \sum_{\substack{k=1 \\ k\neq j}}^{m}(\mathbf{x}_k^T\mathbf{x}_j)\mathbf{y}_k \tag{2.44}$$

式(2.43)右边的第一项代表“期望的”响应 $\mathbf{y}_j$；所以，可以将它看作是实际的响应 $\mathbf{y}$ 的“信号”

部分。第二项 $\mathbf{v}_j$ 是"噪声向量"，它是由关键向量 $\mathbf{x}_j$ 和其他所有存储在记忆中向量的串音产生的。噪声向量 $\mathbf{v}_j$ 是引起回忆误差的根源。

在线性信号空间的情况下，我们可以将一对向量 $\mathbf{x}_j$ 和 $\mathbf{x}_k$ 夹角的余弦定义为 $\mathbf{x}_j$ 和 $\mathbf{x}_k$ 的内积再除以它们各自的欧几里德范数或长度的乘积，表示为

$$\cos(\mathbf{x}_k,\mathbf{x}_j)=\frac{\mathbf{x}_k^T\mathbf{x}_j}{\|\mathbf{x}_k\|\|\mathbf{x}_j\|} \tag{2.45}$$

符号 $\|\mathbf{x}_k\|$ 代表向量 $\mathbf{x}_k$ 的欧几里德范数，定义为 $\mathbf{x}_k$ 的能量的平方根：

$$\|\mathbf{x}_k\|=(\mathbf{x}_k^T\mathbf{x}_k)^{1/2}=E_k^{1/2} \tag{2.46}$$

返回来，注意根据式(2.42)关键向量都被规格化为具有单位能量。因此，我们可以将式(2.45)的定义变为

$$\cos(\mathbf{x}_k,\mathbf{x}_j)=\mathbf{x}_k^T\mathbf{x}_j \tag{2.47}$$

我们可以把式(2.44)中的噪声向量重新定义为

$$\mathbf{v}_j=\sum_{\substack{k=1\\k\neq j}}^{m}\cos(\mathbf{x}_k,\mathbf{x}_j)\mathbf{y}_k \tag{2.48}$$

现在看出，如果关键向量是正交的(也就是说，在欧几里德意义下互相垂直)，那么

$$\cos(\mathbf{x}_k,\mathbf{x}_j)=0,\qquad k\neq j \tag{2.49}$$

81

因此噪声向量 $\mathbf{v}_j$ 为0。在这种情况下，响应 $\mathbf{y}$ 等于 $\mathbf{y}_j$。若关键向量为正交集合，即满足条件

$$\mathbf{x}_k^T\mathbf{x}_j=\begin{cases}1, & k=j\\0, & k\neq j\end{cases} \tag{2.50}$$

那么，联想记忆是完全的。

现在，假定关键向量满足式(2.50)，那么联想记忆的存储能力的限制是多大呢？换句话说，能可靠存储模式的最大数量是多少呢？这一基本问题的答案主要在于记忆矩阵 $\hat{\mathbf{M}}$ 的秩。所谓一个矩阵的秩指的是矩阵中相互不相关的行(列)数。就是说，如果 r 是这样一个 $l\times m$ 维矩阵的秩，那么 $r\leqslant\min(l,m)$。在相关记忆中，记忆矩阵 $\hat{\mathbf{M}}$ 是 $m\times m$ 的矩阵。其中 m 是输入空间的维数。因此，记忆矩阵 $\hat{\mathbf{M}}$ 的秩受维数 m 的限制。我们因而可以正式地说准确无误地存储在相关矩阵记忆中的模式数目决不可能超过输入空间的维数。

在实际生活中，我们通常会发现提供给联想记忆的关键模式既不是正交的，也不是彼此高度分离的。因此由式(2.34)的记忆矩阵所表征的相关矩阵记忆有时会很混乱而且容易产生错误。也就是说，记忆有时会偶尔识别出或联想以前从未见到或相关联的模式。为了说明联想记忆的这一属性，考虑一个关键模式集合

$$\{\mathbf{x}_{\text{key}}\}:\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_q$$

和一个相关记忆模式集合

$$\{\mathbf{y}_{\text{mem}}\}:\mathbf{y}_1,\mathbf{y}_2,\cdots,\mathbf{y}_q$$

为了表示线性信号空间的关键模式密切性，我们引入相似度(community)的概念。我们将模式集合 $\{\mathbf{x}_{\text{key}}\}$ 的相似度定义为集合中任意两个模式 $\mathbf{x}_j$ 和 $\mathbf{x}_k$ 内积 $\mathbf{x}_k^T\mathbf{x}_j$ 的下界。假定 $\hat{\mathbf{M}}$ 表示由对关键模式集合 $\{\mathbf{x}_{\text{key}}\}$ 和与式(2.34)对应的记忆模式集合 $\{\mathbf{y}_{\text{mem}}\}$ 的联想记忆训练导致的记忆矩阵。假定集合 $\{\mathbf{x}_{\text{key}}\}$ 中的每个模式为单位向量(即具有单位能量的向量)，对于 $\{\mathbf{x}_{\text{key}}\}$ 中挑选出的刺激 $\mathbf{x}_j$ 所对应的记忆响应 $\mathbf{y}$ 由式(2.39)给出。进一步假设

$$\mathbf{x}_k^T\mathbf{x}_j \geq \gamma, \qquad 对\ k \neq j \tag{2.51}$$

如下界 γ 足够大，那么记忆不能将响应 $\mathbf{y}$ 与集合 $\{\mathbf{x}_{key}\}$ 中其他任何一个的关键模式区分开来。假如该集合的关键模式具有

$$\mathbf{x}_j = \mathbf{x}_0 + \mathbf{v} \tag{2.52}$$

的形式，其中 $\mathbf{v}$ 是随机向量，记忆很有可能识别出 $\mathbf{x}_0$，并联想出向量 $\mathbf{y}_0$ 而不是原来实际用来训练的模式对；$\mathbf{x}_0$ 和 $\mathbf{y}_0$ 表示以前从未见过的一对模式。这种现象可以称为*动物逻辑*(animal logic)，它完全没有逻辑(Cooper, 1973)。

82

2.12 自适应

在执行某个感兴趣的任务过程中，我们常常发现空间是学习过程中的一个基本的维，而时间是另外一个维。在2.10节讨论的各种学习任务(例如控制、波束形成等)是学习任务的时空性质。从昆虫到人类各种物种都有一种表示经验的时间结构的本能。这种表示使动物可能让它的行为适应它的行为空间中事件的时间结构(Gallistel, 1990)。

从理论上讲，当神经网络处于一个*静态*的环境(即环境的统计特性不随时间变化)，网络对环境的重要统计性质可以在教师监督下进行学习。特别是，网络的突触权值可以通过网络与代表环境的数据集的训练过程而计算得到。一旦训练完成，网络的突触权值就可以捕获环境的基本统计结构，随后就可以"冻结"它们的值。这样一来，学习系统依靠这种或那种形式的记忆，回忆或者利用过去的经验。

然而，环境往往是*非静止的*，即由环境产生的承载信息的信号的统计参数随着时间发生变化。在这种情况下，传统的有监督学习方式是不适合的，因为网络没有相应的必要的方法来跟踪它所处环境的统计变化。为了克服这些不足，希望神经网络最好可以以一种实时的方式，不断地根据输入信号的变化及时调整自由参数。因而*自适应系统*针对每一个不同的输入作出新的响应。换言之，自适应性系统的学习过程永不停息，系统在进行信号处理的同时进行学习。这种形式的学习就叫做*持续学习*(continuous learning)或*飞翔式学习*(learning-on-the-fly)。

*线性自适应滤波器*就是设计用来作为持续学习的。它是建立在线性组合器上的(即在线性模式下运算的单神经元)。尽管它们的结构简单(也许正是因为如此)，它们才被广泛地应用于各种不同领域当中，如雷达、声纳、通信、地震学和生物医疗信号处理。线性自适应性滤波器的理论已经发展到了一个高度成熟阶段(Haykin, 1996; Widrow and Stearns, 1985)。但是非线性自适应性过滤器还未能达到同样的水准[11]。

具有持续学习特性且以神经网络作为它的实现工具，我们必须要解决的问题是：神经网络如何使它的行为适应它的行为空间中输入信号变化的时间结构。解决这一基本问题的一个方面是认识到非静止过程的统计特性通常变化很慢，其过程在一个足够短的时间内考虑为*伪平稳的*。例如：

83

- 产生语音信号的机制在10至30毫秒内可认为是基本平稳的。
- 在几秒内从海洋表面返回的雷达保持基本平稳。
- 对长期大气预报，以分钟计的天气数据可认为基本平稳。
- 在以月和年计的长期趋势中，以天计的股市数据可认为基本平稳。

因此我们可利用随机过程的伪平稳性质，根据输入数据的统计波动在某些固定的时间间

隔内重新训练神经网络以扩展它的应用。例如，这种方法可以适合处理股市数据.

对学习的更好的动态方法，我们可以如下处理：

- 对输入数据挑选足够短的时间窗口，使其可以被认为是伪稳定的，利用该数据训练网络；
- 当收到一个新的数据样本时，丢弃最早的数据样本，向后移动一个时间单位为新样本留出空间，更新窗口。
- 利用更新的数据窗口重新训练网络。
- 在连续的基础上重复这个过程。

从而我们可以利用时序例子使网络经过持续训练在神经网络设计中建立时间结构。根据这个动态方法，神经网络可看作是由线性自适应滤波器推广的非线性自适应滤波器。但是为了非线性自适应滤波器的这个动态方法可行，可用的资源必须足够快使得在一个采样周期内完成所有描述的计算。只有这样滤波器才能和输入变化保持同步。

2.13 学习过程的统计性质

本章的最后部分讨论学习的统计方面。在这里当神经网络通过一个学习算法循环训练时，我们所关心的不是权向量 $\mathbf{w}$ 的演变，而是目标函数 $f(\mathbf{x})$ 和由神经网络所实现的“实际”函数 $F(\mathbf{x}, \mathbf{w})$ 之间的偏差，其中向量 $\mathbf{x}$ 表示输入信号。这种偏差以统计的方式表述。

神经网络只是通过训练可以对一个物理现象或环境的经验知识进行编码的方式之一。“经验知识”这里指标志着环境特征的一组测量。更具体地，考虑一个随机现象的例子，它由包含有一组独立变量的随机向量 $\mathbf{X}$ 和表示一个依赖变量的随机标量 D 描述。随机向量 $\mathbf{X}$ 的元素可以带有它们自己不同的物理含义。依赖变量 D 是一个标量的假设仅仅是为了简化说明而不失一般性。同时假设我们有以 $\{\mathbf{x}_i\}_{i=1}^{N}$ 表示的随机向量 $\mathbf{X}$ 的 N 个实现，以及用 $\{d_i\}_{i=1}^{N}$ 表示的随机标量 D 的一组对应的实现。这些实现(测量)构成了用

84

$$\mathscr{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N} \tag{2.53}$$

表示的训练样本。通常我们不知道 $\mathbf{X}$ 和 D 之间的确切函数关系，所以我们通过提出模型(White, 1989a)

$$D = f(\mathbf{X}) + \varepsilon \tag{2.54}$$

进行讨论，其中 $f(\cdot)$ 是其自变量向量的一个确定性函数，ε 是一个随机期望误差，它代表了我们对 D 和 $\mathbf{X}$ 之间依赖关系的“无知”。由式(2.54)描述的统计模型称作回归模型；它被描述在图 2-20a 中。期望误差 ε 一般是一个带有均值为 0 和正的发生概率的随机变量。在此基础上，图 2-20a 的回归模型有两条有用的性质：

1. 给定任何实现 $\mathbf{x}$，期望误差 ε 的均值为零；即

$$E[\varepsilon \mid \mathbf{x}] = 0 \tag{2.55}$$

其中 E 是数学期望操作符。作为此性质的一个推论，我们可以说回归函数 $f(\mathbf{x})$ 在给定输入 $\mathbf{X}=\mathbf{x}$ 的情况下是模型输出 D 的条件均值，表示为

$$f(\mathbf{x}) = E[D \mid \mathbf{x}] \tag{2.56}$$

这一等式直接根据式(2.55)从式(2.54)得到。

2. 期望误差 ε 与回归函数 $f(\mathbf{X})$ 是不相关的；即

$$E[\varepsilon f(\mathbf{X})] = 0 \tag{2.57}$$

[85] 这个性质就是著名的正交性原理，它说明我们能够通过输入 $\mathbf{X}$ 获取的关于 D 的信息都已被包含进回归函数 $f(\mathbf{X})$ 之中。式(2.57)容易证明如下：

$$E[\varepsilon f(\mathbf{X})] = E[E[\varepsilon f(\mathbf{X}) \mid \mathbf{x}]] = E[f(\mathbf{X})E[\varepsilon \mid \mathbf{x}]] = E[f(\mathbf{X}) \cdot 0] = 0$$

图 2-20a 的回归模型是对随机环境的一个"数学"描述。它的目的是用向量 $\mathbf{X}$ 解释或预测依赖变量 D。图 2-20b 是对应环境的"物理"模型。这第二个基于神经网络的模型的目的是将由训练样本 $\mathscr{T}$ 表示的经验知识编码进对应的一组突触权值向量 $\mathbf{w}$，表示成

$$\mathscr{T} \to \mathbf{w} \tag{2.58}$$

实际上，神经网络提供了一个对图 2-20a 的回归模型的"近似"。令神经网络对输入向量 $\mathbf{x}$ 的实际响应表示为随机变量

$$Y = F(\mathbf{X}, \mathbf{w}) \tag{2.59}$$

其中 $F(\cdot, \mathbf{w})$ 是由神经网络实现的输入－输出函数。给定式(2.53)的训练数据 $\mathscr{T}$，权值向量 $\mathbf{w}$ 通过最小化代价函数

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d_i - F(\mathbf{x}_i, \mathbf{w}))^2 \tag{2.60}$$

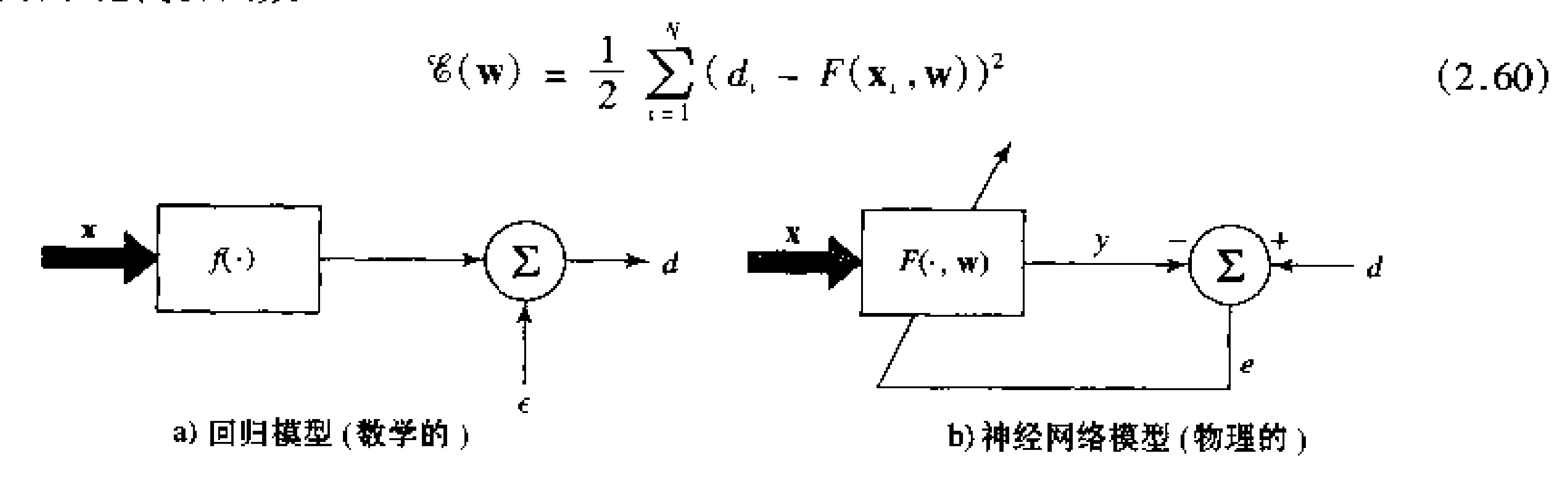


图 2-20　随机环境模型

获得，其中使用因子 1/2 是为了与早先的表示法和后面各章中的表示法保持一致。除了比例因子 1/2 以外，代价函数 $\mathscr{E}(\mathbf{w})$ 是期望响应 d 和神经网络实际响应 y 之间在整个训练数据集 $\mathscr{T}$ 上的平方差。使用式(2.60)作为代价函数意味着采用了"成批"训练，所谓"成批"训练，我们是指对神经网络突触权值的调整是在整个训练样例集上进行的，而不是在单个样例的基础上进行。

令符号 $E_{\mathscr{T}}$ 表示作用于整个训练样本 $\mathscr{T}$ 的均值算子。在均值算子 $E_{\mathscr{T}}$ 作用之下的变量和它们的函数表为 $\mathbf{x}$ 和 d；$(\mathbf{x}, d)$ 对代表训练样本 $\mathscr{T}$ 中的一个例子。与此相对照，统计期望算子 E 作用在随机变量 $\mathbf{X}$ 和 D 的整个总体上，$\mathscr{T}$ 是它的一个子集。算子 E 和 $E_{\mathscr{T}}$ 之间的差异应在下面的讨论中仔细区分。

依据式(2.58)中描述的变换，我们可以交换的使用 $F(\mathbf{x}, \mathbf{w})$ 和 $F(\mathbf{x}, \mathscr{T})$ 并且因此将式(2.60)重写为等价形式

[86]

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}E_{\mathscr{T}}[(d - F(\mathbf{x}, \mathscr{T}))^2] \tag{2.61}$$

对变元 $(d - F(\mathbf{x}, \mathscr{T}))$ 加减 $f(\mathbf{x})$，再利用式(2.54)，我们可写成

$$d - F(\mathbf{x}, \mathscr{T}) = (d - f(\mathbf{x})) + (f(\mathbf{x}) - F(\mathbf{x}, \mathscr{T})) = \varepsilon + (f(\mathbf{x}) - F(\mathbf{x}, \mathscr{T}))$$

通过在式(2.61)中代入此表达式，然后展开项，我们可以将代价函数 $\mathscr{E}(\mathbf{w})$ 重构为等价形式

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}E_{\mathscr{T}}[\varepsilon^2] + \frac{1}{2}E_{\mathscr{T}}[(f(\mathbf{x}) - F(\mathbf{x}, \mathscr{T}))^2] + E_{\mathscr{T}}[\varepsilon(f(\mathbf{x}) - F(\mathbf{x}, \mathscr{T}))] \tag{2.62}$$

然而，式(2.62)右边的最后的一项期望由于下面两个原因而值为0：

- 通过算子 $E_{\mathcal{T}}$ 的解释，依据式(2.57)期望误差 ε 与回归函数 $f(\mathbf{x})$ 是不相关的。
- 期望误差 ε 属于图2-20a的回归模型，而逼近函数 $F(\mathbf{x},\mathbf{w})$ 属于图2-20b的神经网络模型。

从而，式(2.62)化为

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}E_{\mathcal{T}}[\varepsilon^2] + \frac{1}{2}E_{\mathcal{T}}[(f(\mathbf{x}) - F(\mathbf{x},\mathcal{T}))^2] \tag{2.63}$$

式(2.63)右边的第一项是在训练样本$\mathcal{T}$之上计算的期望(回归模型的)误差 ε 的方差。这一项代表*内在误差*，因为它独立于权值向量 $\mathbf{w}$。就最小化关于 $\mathbf{w}$ 的代价函数$\mathscr{E}(\mathbf{w})$而言，它可以被忽略。这样，最小化代价函数$\mathscr{E}(\mathbf{w})$的特定权值向量值 $\mathbf{w}^*$ 也将最小化回归函 $f(\mathbf{x})$ 和逼近函数 $F(\mathbf{x},\mathbf{w})$ 之间的总体平均平方距离。换言之，对 $F(\mathbf{x},\mathbf{w})$ 作为期望响应 d 的预测器的有效性的自然测度定义为

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathbf{w})) = E_{\mathcal{T}}[(f(\mathbf{x}) - F(\mathbf{x},\mathcal{T}))^2] \tag{2.64}$$

这一结果具有根本性的重要意义，因为它为由于使用 $F(\mathbf{x},\mathbf{w})$ 作为对 $f(\mathbf{x})$ 的近似而产生的偏置和方差间的折衷提供了数学基础(Geman et al.,1992)。

偏置/方差困境

回忆式(2.56)的使用，我们可将 $f(\mathbf{x})$ 和 $F(\mathbf{x},\mathbf{w})$ 间的平方距离重新定义如下：

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathbf{w})) = E_{\mathcal{T}}[(E[D \mid \mathbf{X}=\mathbf{x}] - F(\mathbf{x},\mathcal{T}))^2] \tag{2.65}$$

87

这一表达式也可以看作是在整个训练样本$\mathcal{T}$之上计算的回归函数 $f(\mathbf{x}) = E[D \mid \mathbf{X}=\mathbf{x}]$ 和逼近函数 $F(\mathbf{x},\mathbf{w})$ 之间的*估计误差*的平均值。注意，条件均值 $E[D \mid \mathbf{X}=\mathbf{x}]$ 关于训练数据样本$\mathcal{T}$为一个常量期望。进一步我们发现

$$E[D \mid \mathbf{X}=\mathbf{x}] - F(\mathbf{x},\mathcal{T}) = (E[D \mid \mathbf{X}=\mathbf{x}] - E_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]) + (E_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})] - F(\mathbf{x},\mathcal{T}))$$

其中我们只是加上和减去了均值 $E_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]$。通过类似于从式(2.61)中获得式(2.62)那样的方式进行推导，我们可将式(2.65)重写为两项之和(见问题2.22)：

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathcal{T})) = B^2(\mathbf{w}) + V(\mathbf{w}) \tag{2.66}$$

其中 $B(\mathbf{w})$ 和 $V(\mathbf{w})$ 各自定义如下：

$$B(\mathbf{w}) = E_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})] - E[D \mid \mathbf{X}=\mathbf{x}] \tag{2.67}$$

$$V(\mathbf{w}) = E_{\mathcal{T}}[(F(\mathbf{x},\mathcal{T}) - E_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})])^2] \tag{2.68}$$

现在，我们作出两点重要说明：

1. 项 $B(\mathbf{w})$ 是逼近函数 $F(\mathbf{x},\mathcal{T})$ 的平均值对于回归函数 $f(\mathbf{x}) = E[D \mid \mathbf{X}=\mathbf{x}]$ 的偏置。这一项说明由函数 $F(\mathbf{x},\mathbf{w})$ 定义的神经网络不能准确地逼近回归函数 $f(\mathbf{x}) = E[D \mid \mathbf{X}=\mathbf{x}]$。我们因此可以将偏置 $B(\mathbf{w})$ 看作一个*逼近误差*。

2. 项 $V(\mathbf{w})$ 是在整个训练样本$\mathcal{T}$之上测量的逼近函数 $F(\mathbf{x},\mathbf{w})$ 的*方差*。这个项说明包含在训练样本$\mathcal{T}$上中的关于回归函数 $f(\mathbf{x})$ 的信息是不充分的。我们因此可将方差 $V(\mathbf{w})$ 看作是*估计误差*的体现。

图2-21显示目标函数和逼近函数间的关系，以及估计误差也就是偏置和方差是如何积累的。为了取得好的整体性能，逼近函数 $F(\mathbf{x},\mathbf{w}) = F(\mathbf{x},\mathcal{T})$ 的偏置和方差都必须很小才行。

88

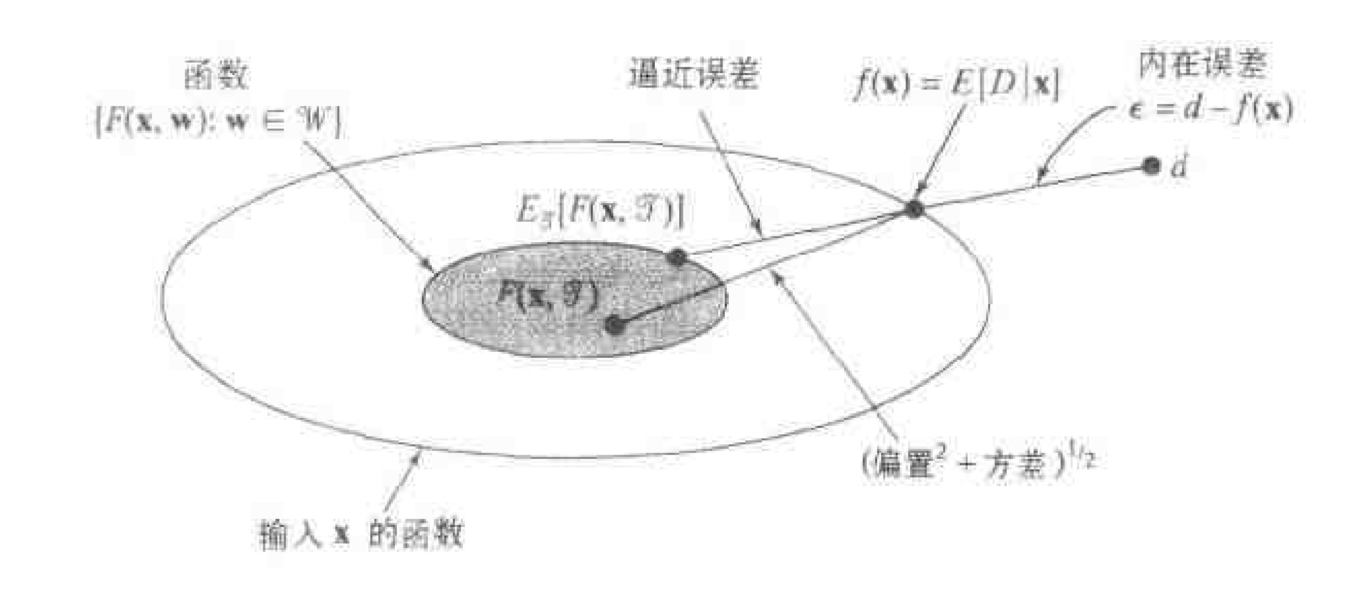


图 2-21 说明求解回归问题的各种误差源

不幸的是，我们发现在以通过例子学习并且训练样本大小是固定的神经网络里，获得小偏置的代价是方差大。对于单个神经网络，只有当训练样本的数量无限时，我们才能指望同时消除偏置和方差两者。于是我们有*偏置/方差困境*，其结果是不可避免的慢收敛性(Geman et al.,1992)。偏置/方差困境可以被避免，如果我们愿意有目的的引入偏置，这样使得消除或大大消减方差成为可能。无需赘言，我们必须保证在网络设计中带入的偏置是无害的。例如，在模式分类的情况下，在下述意义之下偏置被认为是“无害的”，那就是只有当我们试图推断未在预料的分类之中的回归时，偏置才大大提高均方误差。一般来讲，必须为每个特定的应用设计偏置。达到这样目标的一个实用的方法是使用*约束*网络结构，这样的网络结构通常比通用结构有更好的性能。比如，约束以及由此而来的偏置可以使用如下的两种方式采取先验知识的形式嵌入网络设计中：(1)*共享权值*，其中网络的几个突触由一个权值控制，和/或(2)给网络中每个神经元分配*局部接收域*(local receptive field)，就像用多层感知器识别光字符问题中所展示的那样(LeCun et al.,1990a)。这些网络设计问题在 1.7 节中简略讨论过。

2.14 统计学习理论

在这一节里，我们通过描述一种*学习理论*来继续讨论神经网络的统计特性。这种学习理论从数学角度论述如何控制神经网络的推广能力的根本问题。讨论在监督学习下进行。

监督学习模型包括三个相互关联的部分，如图 2-22 所示，从数学角度可抽象如下(Vapnik,1992,1998)：

1. 环境。环境是静态的，提供向量 $\mathbf{x}$，它带有一固定但未知的累积(概率)分布函数。

2. 教师。教师为每个从环境中获得的输入向量 $\mathbf{x}$ 提供一个期望响应 d，其根据是条件累积概率分布函数 $F_{\mathbf{X}}(\mathbf{x}|d)$，它同样是固定但未知的。期望响应 d 和输入向量 $\mathbf{x}$ 的关系为

$$d=f(\mathbf{x},v) \tag{2.69}$$

其中 v 是噪声项，允许教师是“有噪声的”。

3. 学习机器(算法)。学习机器(神经网络)能实

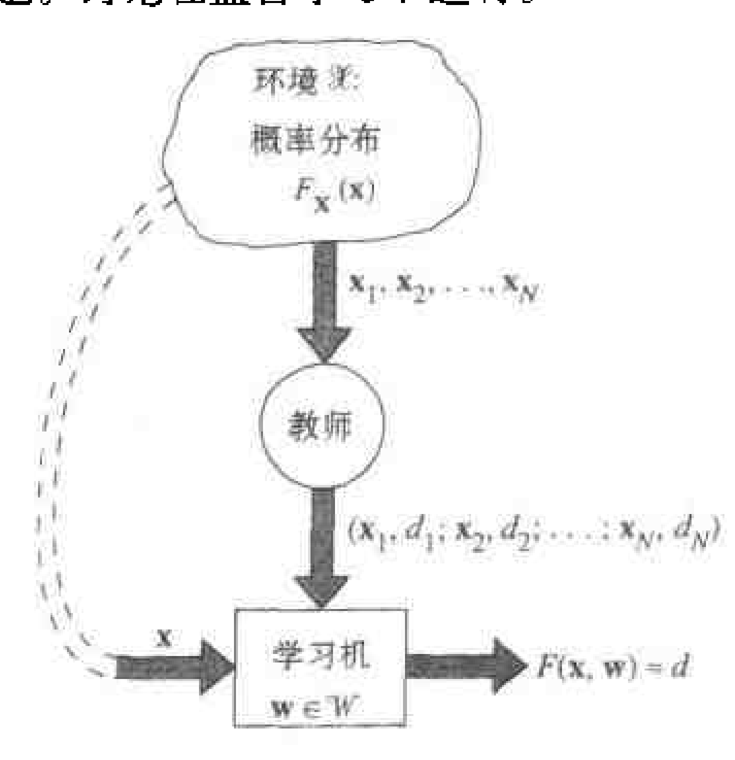


图 2-22 监督学习过程的模型

现一组输入-输出映射函数，描述为

$$y = F(\mathbf{x}, \mathbf{w}) \tag{2.70}$$

89

其中 y 是学习机器对输入 $\mathbf{x}$ 的实际响应，$\mathbf{w}$ 是一组选自参数(突触权值)空间$\mathscr{W}$的自由参数(权值)。

式(2.69)和(2.70)是依据用于完成训练的样例写的。

监督学习问题就是以最优化的方式选择逼近期望响应 d 的特定函数 $F(\mathbf{x}, \mathbf{w})$的问题，这里的"最优化"是以某种统计意义定义的。这种选择本身基于在式(2.53)中描述的 N 个独立同分布的(iid)训练样本，为表示方便重写如下：

$$\mathscr{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$$

每个样例由学习机器以联合累积(概率)分布函数 $F_{\mathbf{X},D}(\mathbf{x}, d)$从$\mathscr{T}$中抽取出来，像其他分布函数一样，$F_{\mathbf{X},D}(\mathbf{x}, d)$同样是固定但未知的。监督学习的可行性取决于这样一个问题：训练样本$\{(\mathbf{x}_i, d_i)\}$是否包含足够的信息来构建具有良好推广性能的学习机器？对这个问题的一个回答在于使用由 Vapnik and Chervonenkis (1971)所开创的工具。特别是，我们通过将监督学习问题视为一个逼近问题开展讨论。这涉及寻找期望函数 $f(\mathbf{x})$的最好逼近函数 $y = F(\mathbf{x}, \mathbf{w})$。

用 $L(d, F(\mathbf{x}, \mathbf{w}))$度量对应于输入向量 $\mathbf{x}$ 的期望响应 d 和由学习机器实际产生的响应 $F(\mathbf{x}, \mathbf{w})$之间的损失或差异。一个普遍的对损失 $L(d, F(\mathbf{x}, \mathbf{w}))$的定义是二次损失函数，它定义为 $d = f(\mathbf{x})$和逼近 $F(\mathbf{x}, \mathbf{w})$之间距离的平方，表示为[12]

$$L(d, F(\mathbf{x}, \mathbf{w})) = (d - F(\mathbf{x}, \mathbf{w}))^2 \tag{2.71}$$

90

式(2.64)的距离平方是对 $L(d, F(\mathbf{x}, \mathbf{w}))$的总体-平均扩展，其平均在所有样例对$(\mathbf{x}, d)$之上计算。

有关统计学习理论的大部分文献都是处理特定的损失。这里讨论的统计学习理论的重要一点是它不严格依赖于损失函数 $L(d, F(\mathbf{x}, \mathbf{w}))$的形式。在本节后面我们将限制讨论具体的损失函数。

损失的期望值由风险泛函

$$R(\mathbf{w}) = \int L(d, F(\mathbf{x}, \mathbf{w}))\, dF_{\mathbf{X},D}(\mathbf{x}, d) \tag{2.72}$$

定义，其中积分是对样例对$(\mathbf{x}, d)$的所有可能值进行的多重积分。监督学习的目标是最小化逼近函数$\{F(\mathbf{x}, \mathbf{w}), \mathbf{w} \in \mathscr{W}\}$之上的风险泛函 $R(\mathbf{w})$。然而，对风险泛函 $R(\mathbf{w})$的求值是复杂的，因为联合累积分布函数 $F_{\mathbf{X},D}(\mathbf{x}, d)$通常是未知的。在监督学习中，惟一能够获取的信息被包含在训练数据集$\mathscr{T}$中。为了克服这一数学上的困难，我们采用经验风险最小化归纳原则(Vapnik, 1982)。这一原则完全依赖于训练数据集$\mathscr{T}$的可用性，这使得它非常适合于神经网络的设计原理。

一些基本定义

在继续讨论之前，我们暂离主题简要介绍一些将要在后面的讨论中使用的基本定义。

依概率收敛　考虑随机变量序列 $a_1, a_2, \cdots, a_N$。如果对任意 $\delta > 0$，概率关系

$$P(|a_N - a_0| > \delta) \xrightarrow{P} 0 \qquad \text{当 } N \to \infty \tag{2.73}$$

成立，意味这一随机变量序列依概率收敛到随机变量 a_0。

上确界和下确界 表示为 $\sup\mathscr{A}$ 的非空的标量集合 $\mathscr{A}$ 的上确界定义为这样的最小标量 x，对于所有 $y\in\mathscr{A}$，有 $x\geq y$。如果没有这样的标量存在，我们说非空集合 $\mathscr{A}$ 的上确界是 ∞。类似地，集合 $\mathscr{A}$ 的下确界，用 $\inf\mathscr{A}$ 表示，被定义为这样的最大标量 x，对于所有 $y\in\mathscr{A}$ 有 $x\leq y$。如果这样的标量不存在，我们说非空集合 $\mathscr{A}$ 的下确界为 $-\infty$。

经验风险泛函 给定训练样本 $\mathscr{T}=\{(\mathbf{x}_i,d_i)\}_{i=1}^N$，经验风险泛函用损失函数 $L(d_i,F(\mathbf{x}_i,\mathbf{w}))$ 定义为

[91]

$$R_{emp}(\mathbf{w})=\frac{1}{N}\sum_{i=1}^{N}L(d_i,F(\mathbf{x}_i,\mathbf{w})) \tag{2.74}$$

严格一致性 考虑函数 $L(d,F(\mathbf{x},\mathbf{w}))$ 的集合 $\mathscr{W}$，$L(d,F(\mathbf{x},\mathbf{w}))$ 的基本分布由联合累积分布函数 $F_{\mathbf{X},D}(\mathbf{x},d)$ 定义。令 $\mathscr{W}(c)$ 为这一函数集合的任意非空子集，使得

$$\mathscr{W}(c)=\left\{\mathbf{w}:\int L(d,F(\mathbf{x},\mathbf{w}))\geq c\right\} \tag{2.75}$$

其中 $c\in(-\infty,+\infty)$。称经验风险泛函是*严格(非平凡)一致的*，如果对任意子集 $\mathscr{W}(c)$，下面的依概率收敛性成立：

$$\inf_{\mathbf{w}\in\mathscr{W}(c)}R_{emp}(\mathbf{w})\xrightarrow{P}\inf_{\mathbf{w}\in\mathscr{W}(c)}R(\mathbf{w}),\qquad \text{当 } N\to\infty \tag{2.76}$$

有了这些定义，我们可以继续讨论 Vapnik 的统计学习理论。

经验风险最小化原则

经验风险最小化原则的基本思想就是处理式(2.74)定义的经验风险泛函 $R_{emp}(\mathbf{w})$。这一新的泛函与式(2.72)的风险泛函的不同之处在于两个期望方式：

1. 它不显式地依赖未知的分布函数 $F_{\mathbf{X},D}(\mathbf{x},d)$。
2. 理论上，它能对权值向量 $\mathbf{w}$ 最小化。

令 $\mathbf{w}_{emp}$ 和 $F(\mathbf{x},\mathbf{w}_{emp})$ 表示最小化式(2.74)中的经验风险泛函 $R_{emp}(\mathbf{w})$ 的权值向量和对应的映射。类似地，令 $\mathbf{w}_o$ 和 $F(\mathbf{x},\mathbf{w}_o)$ 表示最小化式(2.72)中的实际风险泛函 $R(\mathbf{w})$ 的权值向量和对应的映射。$\mathbf{w}_{emp}$ 和 $\mathbf{w}_o$ 都属于权值空间 $\mathscr{W}$。我们现在必须考虑的问题是当用 $R(\mathbf{w}_o)$ 和 $R(\mathbf{w}_{emp})$ 间的差距度量时，近似映射 $F(\mathbf{x},\mathbf{w}_{emp})$ 与期望映射 $F(\mathbf{x},\mathbf{w}_o)$ 相“接近”的条件。

对某一固定 $\mathbf{w}=\mathbf{w}^*$，风险泛函 $R(\mathbf{w}^*)$ 决定了如下定义的随机变量的数学期望：

$$Z_{\mathbf{w}^*}=L(d,F(\mathbf{x},\mathbf{w}^*)) \tag{2.77}$$

相反，经验风险泛函 $R_{emp}(\mathbf{w}^*)$ 是随机变量 $Z_{\mathbf{w}^*}$ 的*经验(算术)平均值*。根据概率论的主要理论之一的*大数定律*，在一般情形下，我们发现当训练样本 $\mathscr{T}$ 无穷大时，随机变量 $Z_{\mathbf{w}^*}$ 的经验均值收敛于它的期望值。这一事实为使用经验风险泛函 $R_{emp}(\mathbf{w})$ 来代替风险泛函 $R(\mathbf{w})$ 提供了理论证据。然而，正是由于 $Z_{\mathbf{w}^*}$ 的经验均值收敛于它的期望值，就没有理由指望最小化经验风险泛函 $R_{emp}(\mathbf{w})$ 的权值向量 $\mathbf{w}_{emp}$ 同样会最小化风险泛函 $R(\mathbf{w})$。

[92]

我们可以按下述的方法进行，以近似的方式满足这一需要。如果经验风险泛函 $R_{emp}(\mathbf{w})$ 按 $\mathbf{w}$ 以某一精度 ε 一致地逼近原始风险泛函 $R(\mathbf{w})$，那么 $R_{emp}(\mathbf{w})$ 的最小值对 $R(\mathbf{w})$ 的最小值的偏离不超过 2ε。从形式上说，这意味着我们必须施加一个严格条件使得对任何 $\mathbf{w}\in\mathscr{W}$ 和

$\varepsilon > 0$，概率关系

$$P(\sup_{\mathbf{w}} |R(\mathbf{w}) - R_{emp}(\mathbf{w})| > \varepsilon) \to 0 \qquad 当 N \to \infty \tag{2.78}$$

成立(Vapnik,1982)。当满足式(2.78)时，我们说出现经验平均风险的权值向量 $\mathbf{w}$ 到期望值的一致收敛。等价地，如果对任何指定的精度 ε，我们能对某 $\alpha > 0$ 确定不等式

$$P(\sup_{\mathbf{w}} |R(\mathbf{w}) - R_{emp}(\mathbf{w})| > \varepsilon) < \alpha \tag{2.79}$$

那么，结果是如下的不等式也成立：

$$P(R(\mathbf{w}_{emp}) - R(\mathbf{w}_o) > 2\varepsilon) < \alpha \tag{2.80}$$

换言之，如果条件(2.79)成立，那么至少以概率 $1-\alpha$，最小化经验风险泛函 $R_{emp}(\mathbf{w})$ 的解 $F(\mathbf{x}, \mathbf{w}_{emp})$ 给出的实际风险 $R(\mathbf{w}_{emp})$ 与真正的最小化可能实际风险 $R(\mathbf{w}_o)$ 的偏差不会超过 2ε。确实，条件(2.79)意味着如下的两个不等式以概率 $1-\alpha$ 同时得到满足(Vapnik,1982)：

$$R(\mathbf{w}_{emp}) - R_{emp}(\mathbf{w}_{emp}) < \varepsilon \tag{2.81}$$

$$R_{emp}(\mathbf{w}_o) - R(\mathbf{w}_o) < \varepsilon \tag{2.82}$$

这两个不等式分别定义了真实风险和经验风险泛函在 $\mathbf{w} = \mathbf{w}_{emp}$ 和 $\mathbf{w} = \mathbf{w}_o$ 的差异。此外，由于 $\mathbf{w}_{emp}$ 和 $\mathbf{w}_o$ 分别为 $R_{emp}(\mathbf{w})$ 和 $R(\mathbf{w})$ 的最小点，于是有

$$R_{emp}(\mathbf{w}_{emp}) \leqslant R_{emp}(\mathbf{w}_o) \tag{2.83}$$

通过将不等式(2.81)和(2.82)相加，然后使用(2.83)，我们可以重写不等式

$$R(\mathbf{w}_{emp}) - R(\mathbf{w}_o) < 2\varepsilon \tag{2.84}$$

同样，由于不等式(2.81)和(2.82)同时以概率$(1-\alpha)$得到满足，所以不等式(2.84)也一样。我们也可以表达为不等式

$$R(\mathbf{w}_{emp}) - R(\mathbf{w}_o) > 2\varepsilon$$

以概率 α 成立，这是对(2.80)的重述。

现在，我们可以对经验风险最小化原则从三个相互联系的部分做一个形式化的陈述(Vapnik,1982,1998)：

1. 代替风险泛函 $R(\mathbf{w})$，构建经验风险泛函

$$R_{emp}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L(d_i, F(\mathbf{x}_i, \mathbf{w}))$$

它基于独立同分布的样本训练集$(\mathbf{x}_i, d_i)$，$i = 1, 2, \cdots, N$

93

2. 令 $\mathbf{w}_{emp}$ 表示在权值空间$\mathscr{W}$上最小化经验风险泛函的权值向量。那么只要经验风险泛函 $R_{emp}(\mathbf{w})$一致收敛于实际风险泛函 $R(\mathbf{w}_{emp})$，当训练样本的数量 N 趋于无穷大时，$R_{emp}(\mathbf{w})$ 依概率收敛到实际风险 $R(\mathbf{w})$，$\mathbf{w} \in \mathscr{W}$ 的最小可能值。

3. 由

$$P(\sup_{\mathbf{w}\in\mathscr{W}} |R(\mathbf{w}) - R_{emp}(\mathbf{w})| > \varepsilon) \to 0 \qquad 当 N \to \infty$$

定义的一致收敛性是经验风险最小化原则一致性的充分必要条件。

为了对这一重要原则有一个自然解释，我们给出如下分析。在学习机器开始训练之前，所有逼近函数都是等可能的。随着学习机器训练的进行，与训练数据集$\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$相符的那些逼近函数 $F(\mathbf{x}_i, \mathbf{w})$ 的可能性增加了。当训练数据集的数量 N 增长时，输入空间因此密集起来，经验风险泛函 $R_{emp}(\mathbf{w})$ 的最小点依概率收敛到真实风险泛函 $R(\mathbf{w})$ 的最小点。

VC维

经验风险泛函 $R_{emp}(\mathbf{w})$到实际风险泛函 $R(\mathbf{w})$的一致收敛性理论包括收敛速度的界，它们基于称为 Vapnik-Chervonenkis 维(或简称 VC 维)的重要参数，其名称是为了纪念它的创立者 Vapnik 和 Chervonenkis。VC 维是对由学习机器实现的分类函数族的容量或表示能力的测度。

为了以适合于我们目的的方式描述 VC 维的概念，考虑二值模式分类问题，为此期望响应写作 $d\in\{0,1\}$。我们使用术语二分(dichotomy)来指二值分类函数或判定规则。令$\mathscr{F}$表示由学习机器实现的二分的总体，即

$$\mathscr{F} = \{F(\mathbf{x},\mathbf{w}):\mathbf{w}\in\mathscr{W}, F:\mathbb{R}^m\mathscr{W}\rightarrow\{0,1\}\} \tag{2.85}$$

令$\mathscr{L}$表示输入向量的 m - 维空间$\mathscr{X}$中的 N 个点的集合，即

$$\mathscr{L} = \{\mathbf{x}_i\in\mathscr{X}; i = 1,2,\cdots,N\} \tag{2.86}$$

一个由学习机器实现的二分将$\mathscr{L}$分割为两个不相交的子集$\mathscr{L}_0$ 和$\mathscr{L}_1$，使得我们有

$$F(\mathbf{x},\mathbf{w}) = \begin{cases}0, 对\ \mathbf{x}\in\mathscr{L}_0\\ 1, 对\ \mathbf{x}\in\mathscr{L}_1\end{cases} \tag{2.87}$$

令 $\Delta_{\mathscr{F}}(\mathscr{L})$表示能由学习机器实现的不同二分的数量，$\Delta_{\mathscr{F}}(l)$表示在所有$|\mathscr{L}| = l$ 的$\mathscr{L}$上$\Delta_{\mathscr{F}}(\mathscr{L})$的最大值，其中$|\mathscr{L}|$是$\mathscr{L}$的元素的数量。我们说$\mathscr{L}$被$\mathscr{F}$分散，如果 $\Delta_{\mathscr{F}}(\mathscr{L}) = 2^{|\mathscr{L}|}$，即如果$\mathscr{L}$的所有的二分都能被$\mathscr{F}$中的函数所产生。我们称 $\Delta_{\mathscr{F}}(l)$为增长函数。

94

例2.1 图 2-23 显示了包含 4 个点 $\mathbf{x}_1$，$\mathbf{x}_2$，$\mathbf{x}_3$ 和 $\mathbf{x}_4$ 的一个二维输入空间$\mathscr{X}$。图中所指示的函数 F_0 和 F_1 的判定边界分别对应于正确的类(假设) 0 和 1。从图 2-23 中，我们看出函数 F_0 导出二分

$$\mathscr{D}_0 = \{\mathscr{S}_0 = \{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_4\},\mathscr{S}_1 = \{\mathbf{x}_3\}\}$$

另一方面，函数 F_1 导出二分

$$\mathscr{D}_1 = \{\mathscr{S}_0 = \{\mathbf{x}_1,\mathbf{x}_2\},\mathscr{S}_1 = \{\mathbf{x}_3,\mathbf{x}_4\}\}$$

对于包含 4 个点的集合$\mathscr{S}$，基$|\mathscr{S}| = 4$。从而

$$\Delta_{\mathscr{F}}(\mathscr{S}) = 2^4 = 16$$

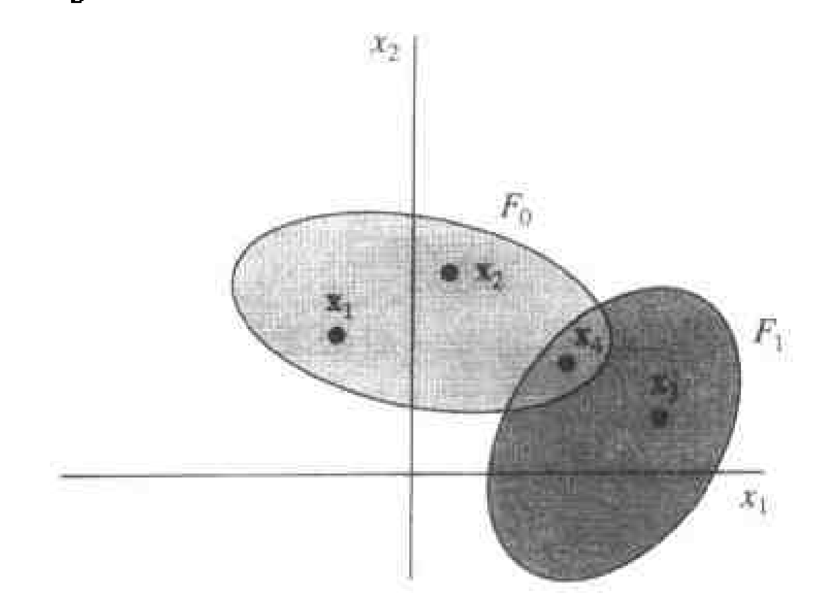


图 2-23 例 2-1 的图形

■

回到式(2.85)中的二分总体$\mathscr{F}$及式(2.86)中的对应点集$\mathscr{L}$所勾画的一般讨论，我们现在可以正式地定义 VC 维如下(Vapnik and Chervonenkis, 1971; Kearns and Vazirani, 1994; Vidyasagar, 1997; Vapnik, 1998)：

二分总体$\mathscr{F}$的 VC 维是被$\mathscr{F}$所分散的最大集合$\mathscr{L}$的基数。

换言之，$\mathscr{F}$的 VC 维(写作 VCdim($\mathscr{F}$))是使 $\Delta_{\mathscr{F}}(N) = 2^N$ 的最大 N。用更熟悉的话说，分类函数集$\{F(\mathbf{x},\mathbf{w}):\mathbf{w}\in\mathscr{W}\}$的 VC 维是能被机器学习的训练样本的最大数量，这种学习对于分类函数所有可能的二分标记是无错误的。

例2.2 考虑输入向量的 m 维空间$\mathscr{X}$中的一个简单判定规则，它由

$$\mathscr{F}: y = \varphi(\mathbf{w}^T\mathbf{x} + b) \tag{2.88}$$

[95]

描述，其中 $\mathbf{x}$ 是一个 m 维权值向量，b 是偏置。激活函数 φ 是一个阈值函数，即

$$\varphi(v) = \begin{cases} 1, & v \geq 0 \\ 0, & v < 0 \end{cases}$$

式(2.88)中的判定规则的 VC 维给出如下：

$$\text{VCdim}(\mathscr{F}) = m + 1 \tag{2.89}$$

为了说明这一结论，考虑图 2-24 中所描绘的二维输入空间(即 $m = 2$)的情况。在图 2-24a 中，我们有 $\mathbf{x}_1$，$\mathbf{x}_2$ 和 $\mathbf{x}_3$ 三个点。对这三个点的三种可能标记包括在图 2-24a 中，从中我们很容易看到最多三条线就能分散这些点。在图 2-24b 中，我们有点 $\mathbf{x}_1$，$\mathbf{x}_2$，$\mathbf{x}_3$ 和 $\mathbf{x}_4$，点 $\mathbf{x}_2$ 和 $\mathbf{x}_3$ 标记为 0，点 $\mathbf{x}_1$ 和 $\mathbf{x}_4$ 标记为 1。可是这一次，我们看到点 $\mathbf{x}_1$ 和 $\mathbf{x}_4$ 不能用一条直线与点 $\mathbf{x}_2$ 和 $\mathbf{x}_3$ 中分散开来。式(2.88)中所描述的 $m = 2$ 判定规则的 VC 维因此为 3，这是与式(2.89)相符的。

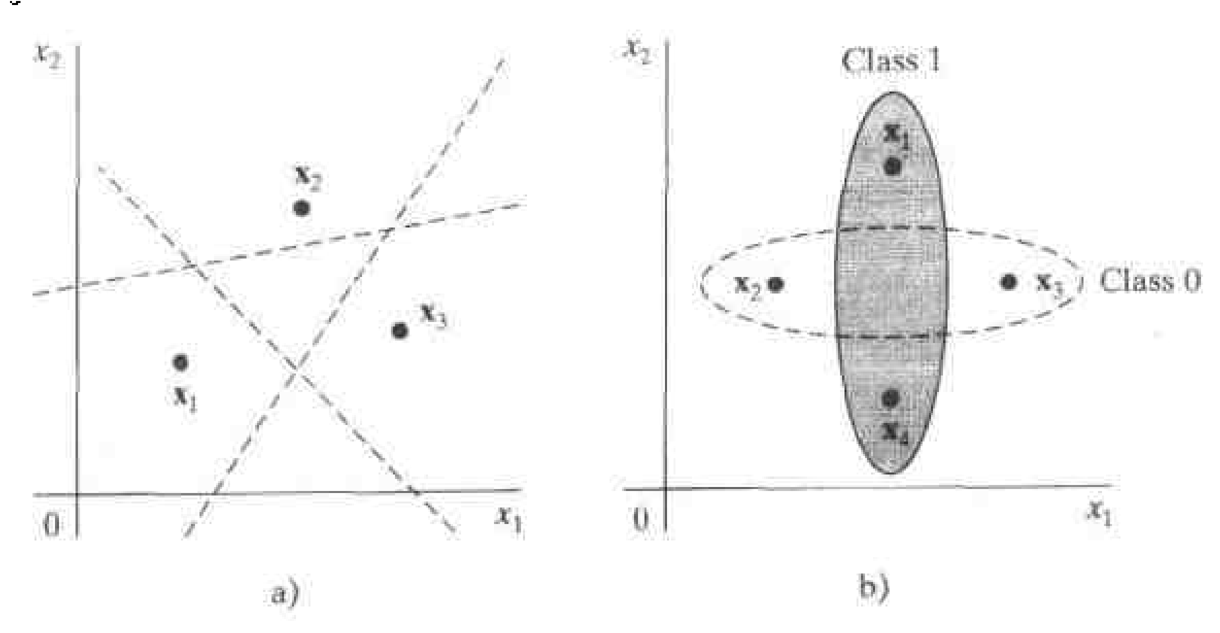


图 2-24　例 2.2 的一对两维数据分布

例 2.3　用 VC 维来度量分类(指示器)函数集的容量，我们也许可能期望带有很多自由参数的学习机器会有高的 VC 维，而带有少数的自由参数的学习机器会有低的 VC 维。我们现在举一个这一说法的反例[13]。

考虑只有一个参数的指示函数族，定义如下：

$$f(x, a) = \text{sgn}(\sin(ax)), \qquad a \in \mathbb{R}$$

其中 sgn(·)是符号函数。假设我们选择任意数 N，需要做的是找到能被分散的 N 个点。这一要求通过选择

$$x_i = 10^{-i}, \qquad i = 1, 2, \cdots, N$$

而被函数集 $f(x, a)$ 所满足。为了将这些数据分成由序列

$$d_1, d_2, \cdots, d_N, \qquad d_i \in \{-1, 1\}$$

[96]

所确定的两类，我们根据公式

$$a = \pi\left(1 + \sum_{i=1}^{N} \frac{(1 - d_i)10^i}{2}\right)$$

来选择参数 a 就足够了。这样我们得出结论，带有单一自由参数 a 的指示函数族 $f(x, a)$ 的 VC 维是无穷的。

VC维的重要性及其估计

VC维是一个与几何概念的维没有关系的纯粹组合概念。它在统计学习理论中扮演着一个中心的角色，这将在后面两小节提供的材料中看出来。从设计的观点看，VC维也是重要的。粗略地说，为了可靠地学习一个类所需要的样本的数量正比于那个类的VC维。因此，对VC维的估计需要首先关注。

在一些情况下，VC维由神经网络的自由参数决定。然而在大多数实际情况下，很难通过分析的手段计算VC维。虽然如此，神经网络的VC维的界经常是容易处理的。这时，下面的两个结论具有特殊意义[14]：

1. 令$\mathcal{N}$表示由神经元构成的任意前馈网络，阈值(Heaviside)激活函数为

$$\varphi(v)=\begin{cases}1, & v\geq 0\\ 0, & v<0\end{cases}$$

$\mathcal{N}$的VC维为$O(W\log W)$，其中W是网络中自由参数的总数。

这第一个结论归功于Cover(1968)和Baum and Haussler(1989)。

2. 令$\mathcal{N}$表示一个多层前馈网络，其神经元使用一个sigmoid激活函数

$$\varphi(v)=\frac{1}{1+\exp(-v)}$$

$\mathcal{N}$的VC维为$O(W^2)$，其中W是网络中自由参数的总数。

这第二条结论归功于Koiran and Sontag(1996)。他们得出这一结论是通过首先证明包含两类神经元(一类是线性的，另一类使用阈值激活函数)的网络已有了正比于W^2的VC维。这是个相当令人惊异的结论，因为像在例2.2中看到的那样一个纯线性网络有正比于W的VC维，而根据第一个结论一个纯阈值神经网络有一个正比于$W\log W$的VC维。接着，通过求助于两种近似就得到关于sigmoid神经网络的理想结论。第一，具有阈值激活函数的神经元由具有大突触权值的sigmoid式神经元近似。第二，线性神经元由具有小突触权值的sigmoid神经元近似。

97

这里需注意的重要一点是，多层前馈网络具有有限的VC维。

学习机器推广能力的构造性自由分布界

讨论进行到这里，我们发现考虑二值模式分类的具体情况是有益的。这种分类的期望响应定义为$d\in\{0,1\}$。相应的损失函数只有如下两个可能值：

$$L(d,F(\mathbf{x},\mathbf{w}))=\begin{cases}0, & 若\ F(\mathbf{x},\mathbf{w})=d\\ 1, & 其他\end{cases}\tag{2.90}$$

在这些条件下，分别在式(2.72)和(2.74)中定义的风险泛函$R(\mathbf{w})$和经验风险泛函$R_{emp}(\mathbf{w})$得到如下解释：

- 风险泛函$R(\mathbf{w})$是分类错误的概率(即误差率)，表示为$P(\mathbf{w})$。
- 经验风险泛函$R_{emp}(\mathbf{w})$是训练误差(即训练阶段发生错误的频率)，表示为$\nu(\mathbf{w})$。

现在，根据大数定律(Gray & Davisson,1986)，一个事件发生的经验频率几乎一定收敛于那一事件的实际概率，只要试验(假设是独立同分布的)的数目趋于无穷大。在这里讨论的情

况下，这一结论意味着对任何权值向量 $\mathbf{w}$，它不依赖于训练集，以及对任何精度 $\epsilon>0$，下面条件成立：

$$P(|P(\mathbf{w})-\nu(\mathbf{w})|>\epsilon)\to 0 \qquad \text{当 } N\to\infty \tag{2.91}$$

其中，N 是训练集的大小。然而，请注意，条件(2.91)并不意味着最小化训练误差 $\nu(\mathbf{w})$的分类规则(即一个特定的权值向量 $\mathbf{w}$)也会最小化分类误差概率 $P(\mathbf{w})$。对于一个具有充分大的数量 N 的训练集来说，$\nu(\mathbf{w})$与 $P(\mathbf{w})$的接近服从一个更强的条件，它规定下面的条件对任何 $\epsilon>0$ 都成立(Vapnik,1982)：

$$P(\sup_{w}|P(\mathbf{w})-\nu(\mathbf{w})|>\epsilon)\to 0 \qquad \text{当 } N\to\infty \tag{2.92}$$

在这种情况下，我们就说训练误差频率到 $\nu(\mathbf{w})=P(\mathbf{w})$的概率一致收敛。

VC 维的概念在一致收敛的速度上提供了一个界。特别，对于 VC 维为 h 的分类函数集，下面的不等式成立(Vapnik,1982,1998)：

$$P(\sup_{w}|P(\mathbf{w})-\nu(\mathbf{w})|>\epsilon)<\left(\frac{2eN}{h}\right)^{h}\exp(-\epsilon^{2}N) \tag{2.93}$$

其中 N 是训练样本的大小，e 是自然对数的底。为了获得一致收敛性我们希望不等式(2.93)的右边对于大 N 会变小。因子 $\exp(-\epsilon^{2}N)$在这一方面是有帮助的，因为它随着 N 的上升而指数下降。剩下的因子$(2eN/h)^{h}$ 代表函数族$\mathscr{F}=\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathscr{W}\}$的增长函数 $\Delta_{\mathscr{F}}(l)$当 $l\geq h\geq 1$时的界，这由 Sauer 引理[15]得到。只要这一函数不要增长太快，右边会随着 N 趋于无穷大而趋于零；如果 VC 维 h 是有限的，这一要求就得到满足。换言之，有限的 VC 维是经验风险最小化原则的一致收敛性的充分必要条件。如果输入空间$\mathscr{X}$有有限的基数，任何二分标记族$\mathscr{F}$都会有关于$\mathscr{X}$的有限 VC 维，虽然逆命题并不一定成立。

令 α 表示事件

$$\sup_{\mathbf{w}}|P(\mathbf{w})-\nu(\mathbf{w})|\geq\epsilon$$

发生的概率。那么，以概率 $1-\alpha$，我们可以说对所有权值向量 $\mathbf{w}\in\mathscr{W}$，下面的不等式成立：

$$P(\mathbf{w})<\nu(\mathbf{w})+\epsilon \tag{2.94}$$

使用式(2.93)中描述的界和概率 α 的定义，我们可以置

$$\alpha=\left(\frac{2eN}{h}\right)^{h}\exp(-\epsilon^{2}N) \tag{2.95}$$

令 $\epsilon_0(N,h,\alpha)$表示满足式(2.95)的特殊值。由此，我们很容易得到下面的重要结论(Vapnik,1992)：

$$\epsilon_0(N,h,\alpha)=\sqrt{\frac{h}{N}\left[\log\left(\frac{2N}{h}\right)+1\right]-\frac{1}{N}\log\alpha} \tag{2.96}$$

我们称 $\epsilon_0(N,h,\alpha)$为置信区间，其值取决于训练样本的大小 N 以及 VC 维 h 和概率 α。

式(2.93)中以 $\epsilon=\epsilon_0(N,h,\alpha)$描述的界在最坏的情况 $P(\mathbf{w})=1/2$ 下获得，但不幸的是并非对小的 $P(\mathbf{w})$成立，而这是实际中感兴趣的情况。对于小的 $P(\mathbf{w})$，通过考虑如下修改不等式(2.93)可获得更有用的界(Vapnik,1982,1998)：

$$P\left(\sup_{\mathbf{w}}\frac{|P(\mathbf{w})-\nu(\mathbf{w})|}{\sqrt{P(\mathbf{w})}}>\epsilon\right)<\left(\frac{2eN}{h}\right)^{h}\exp\left(-\frac{\epsilon^{2}N}{4}\right) \tag{2.97}$$

在文献中，对式(2.97)中的界报导的不同结果，取决于使用不等式的哪个特定形式来推导。不过，它们都有一个相似的形式。从(2.97)推出，用概率 $1-\alpha$，并且同时对于所有 $\mathbf{w}\in\mathscr{W}$有

98

(Vapnik, 1992, 1998)

99

$$P(\mathbf{w}) \le \nu(\mathbf{w}) + \varepsilon_1(N,h,\alpha,\nu) \tag{2.98}$$

其中 $\varepsilon_1(N,h,\alpha,\nu)$是一个新的置信区间，它是用前一个置信区间 $\varepsilon_0(N,h,\alpha)$来定义的，如下(参看习题2.25)：

$$\varepsilon_1(N,h,\alpha,\nu) = 2\varepsilon_0^2(N,h,\alpha)\left(1+\sqrt{1+\frac{\nu(\mathbf{w})}{\varepsilon_0^2(N,h,\alpha)}}\right) \tag{2.99}$$

这第二个置信区间取决于训练误差 $\nu(\mathbf{w})$。对于 $\nu(\mathbf{w})=0$，它归为特殊形式

$$\varepsilon_1(N,h,\alpha,0) = 4\varepsilon_0^2(N,h,\alpha) \tag{2.100}$$

我们现在可以总结一下已经为一致收敛速度推导出的两个界：

1. 一般情况下，我们有如下一致收敛速度的界：

$$P(\mathbf{w}) \le \nu(\mathbf{w}) + \varepsilon_1(N,h,\alpha,\nu)$$

其中 $\varepsilon_1(N,h,\alpha,\nu)$如式(2.99)中的定义。

2. 对于接近于0的小的训练误差 $\nu(\mathbf{w})$，我们有

$$P(\mathbf{w}) \le \nu(\mathbf{w}) + 4\varepsilon_0^2(N,h,\alpha)$$

它为真实情况中的学习提供了一个相当精确的界。

3. 对于接近于1的大训练误差 $\nu(\mathbf{w})$，我们有界

$$P(\mathbf{w}) \le \nu(\mathbf{w}) + \varepsilon_0(N,h,\alpha)$$

结构风险最小化

训练误差是具有某一权值向量的学习机器在训练阶段所犯错误的频率。相似地，泛化误差被定义为当用机器以前没有见过的样本测试它时所犯错误的频率。这里假设测试数据是从与训练数据相同的总体抽取得到的。令这两种误差分别表示为 $\nu_{\text{train}}(\mathbf{w})$和 $\nu_{\text{gene}}(\mathbf{w})$。注意 $\nu_{\text{train}}(\mathbf{w})$与前面小节中所用的 $\nu(\mathbf{w})$相同；那里我们是用 $\nu(\mathbf{w})$来简化表示法。令 h 为分类函数族$\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathcal{W}\}$关于输入空间$\mathcal{X}$的VC维。那么，依据关于一致收敛速度的理论，我们可以说以概率$1-\alpha$，对于训练样本的数量$N>h$，以及同时对所有的分类函数$F(\mathbf{x},\mathbf{w})$，泛化误差 $\nu_{\text{gene}}(\mathbf{w})$比保证风险小，保证风险定义为两个竞争项的和(Vapnik, 1992, 1998)

$$\nu_{\text{guarant}}(\mathbf{w}) = \nu_{\text{train}}(\mathbf{w}) + \varepsilon_1(N,h,\alpha,\nu_{\text{train}}) \tag{2.101}$$

其中置信区间 $\varepsilon_1(N,h,\alpha,\nu_{\text{train}})$本身由式(2.99)定义。对于固定数量的训练样本 N，训练误差随着容量或VC维 h 的增加而单调递减，而置信区间单调递增。因此，保证风险和泛化误差都经历最小值。这些趋势在图2-25中以普通的方式展示出来。在达到最小点之前，机器容量对于训练细节的数量是太小了，在这个意义上，说学习问题是过定的(overdetermined)。超过最小点后，学习问题是

100

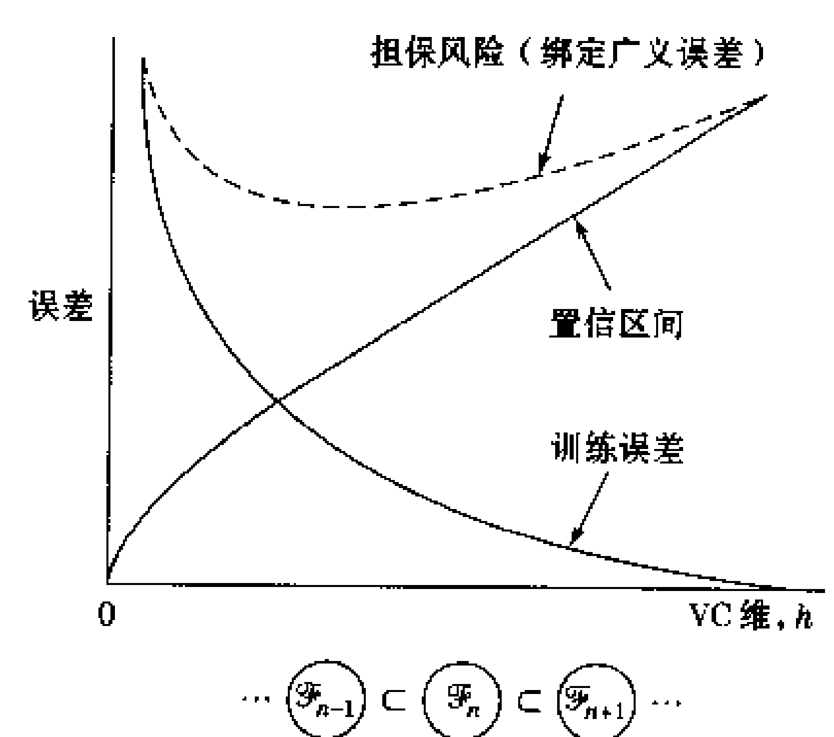


图2-25 训练误差、置信区间和担保风险之间的关系图示

大定的，因为机器容量对于训练数据是太大了。

所以、解决监督学习问题的挑战就是通过使机器容量与用于目前问题的训练数据的有效数量相匹配。结构风险最小化方法通过使学习机器的VC维成为一个控制变量来提供一个归纳过程以达到上述目标(Vapnik,1992,1998)。具体地说，考虑模式分类器的集合$\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathscr{W}\}$，并定义$n$个这样机器的嵌套结构

$$\mathscr{F}_k=\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathscr{W}_k\},\qquad k=1,2,\cdots,n \tag{2.102}$$

使得我们有(参看图2-25)

$$\mathscr{F}_1\subset\mathscr{F}_2\subset\cdots\subset\mathscr{F}_n \tag{2.103}$$

其中符号$\subset$意指“包含于”。相应地，各个模式分类器的VC维满足条件

$$h_1\leqslant h_2\leqslant\cdots\leqslant h_n \tag{2.104}$$

这意味着每个模式分类器的VC维是有限的。所以，结构风险最小化方法可如下进行：

- 对每个模式分类器，最小化经验风险(即训练误差)。
- 确定具有最小保证风险的模式分类器$\mathscr{F}^*$；这一特殊机器提供相互竞争的训练误差(即对训练数据近似的质量)和置信区间(即逼近函数的复杂性)之间的最好的折衷。

我们的目标就是找到一个网络结构，使得能以训练误差最小可能增加为代价来换取VC维的降低。

结构风险最小化原则能以多种方法实现。例如，我们可以通过改变隐藏神经元的个数来改变VC维h。特别是，我们评估全连接的多层前馈网络的总体，该网络中一个隐藏层的神经元数量以单调的方式增加。结构风险最小化原则表明，这一总体中最好的网络是保证风险最小的那一个。 101

VC维不仅是结构风险最小化原则的核心，而且也是一个称为可能近似正确(PAC)的同等强大的学习模型的核心。在下节讨论后一个模型，以此来结束本章处理学习的概率和统计方面的最后一部分。

2.15 可能近似正确的学习模型

可能近似正确(probably approximately correct,PAC)的学习模型归功于Valiant(1984)。顾名思义，PAC模型为二值分类系统中的学习和推广的研究提供了概率框架。它与监督学习紧密相关。

我们从环境$\mathscr{X}$入手。一个$\mathscr{X}$的集合称为概念(concept)，$\mathscr{X}$的子集的集合称为概念类(concept class)。一个概念的例(example)是具有一个类标签的论域中的一个对象(object)。如果该例是概念的一个成员，我们称之为*正例*(positive example)；如果该对象不是概念的一个成员，我们称之为*反例*(negative example)。提供例的概念称作*目标概念*。对于一个目标概念c，我们需要长度为N的训练数据的序列，由

$$\mathscr{T}=\{(\mathbf{x}_i,c(\mathbf{x}_i))\}_{i=1}^{N} \tag{2.105}$$

表示，其中可能包含重复的例。例$\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$是根据某个固定但未知的概率分布从环境$\mathscr{X}$中随机抽取出来的。式(2.105)中有如下两点值得注意：

- 目标概念$c(\mathbf{x}_i)$被当作从$\mathscr{X}$到$\{0,1\}$的一个函数，而且$c(\mathbf{x}_i)$假定是未知的。
- 这些例通常被认为是统计独立的，这意味着任何两个例(比如说$\mathbf{x}_i$和$\mathbf{x}_j$)的联合概率密度函数等于它们各自概率密度函数的积。

在我们前述术语的上下文中，环境$\mathscr{X}$可以等同于神经网络的输入空间，目标概念等同于网络的期望响应。

从环境$\mathscr{X}$中导出的概念的集合称作概念空间$\mathscr{L}$。例如，概念空间可能会包含"字母A"，"字母B"等等。这些概念中的每一个可能以不同的编码生成一个正例集合和一个反例集合。102 然而在监督学习的框架中，我们有另一组概念。一个学习机器典型地代表一个函数集，其中的每个函数对应一个特定的状态。例如，机器可能被设计成识别"字母A"，"字母B"等。由学习机器决定的所有函数(即概念)的集合称为假设空间$\mathscr{G}$。假设空间可能等于或不等于概念空间。在某种意义上，概念空间和假设空间的含义可以分别与在前一节所讨论的函数$f(\mathbf{x})$和逼近函数$F(\mathbf{x},\mathbf{w})$相类比。

那么，假定我们有一个目标概念$c(\mathbf{x})\in\mathscr{C}$，它只取值0或1。我们希望由一个神经网络来学会这一概念，这个神经网络由式(2.105)定义的数据集$\mathscr{T}$训练。令$g(\mathbf{x})\in\mathscr{G}$表示与这个训练得到的输入－输出映射相对应的假设。评价学习过程是否成功的方法之一是度量假设$g(\mathbf{x})$离目标概念$c(\mathbf{x})$有多接近。如果$g(\mathbf{x})\neq c(\mathbf{x})$，自然有误差发生。产生误差的原因是我们试图以一个函数有限的可用信息为基础来学习这个函数。训练误差的概念定义为

$$\nu_{\text{train}} = P(\mathbf{x}\in\mathscr{X}; g(\mathbf{x})\neq c(\mathbf{x})) \tag{2.106}$$

这个式中的概率分布必需与用于生成样本的分布一样。PAC学习的目标就是确保ν_{train}通常是小的。可以用于学习算法的域由训练样本$\mathscr{T}$的大小N控制。另外，为学习算法提供了两个控制参数：

- *误差参数* $\varepsilon\in(0,1]$。这个参数指定在假设$g(\mathbf{x})$对目标概念$c(\mathbf{x})$的一个良好近似中所允许的误差。
- *置信参数* $\delta\in(0,1]$。这第二个参数控制构建一个良好逼近的可能性。

我们从而可以将PAC学习模型看作如图2-26中描绘的那样。

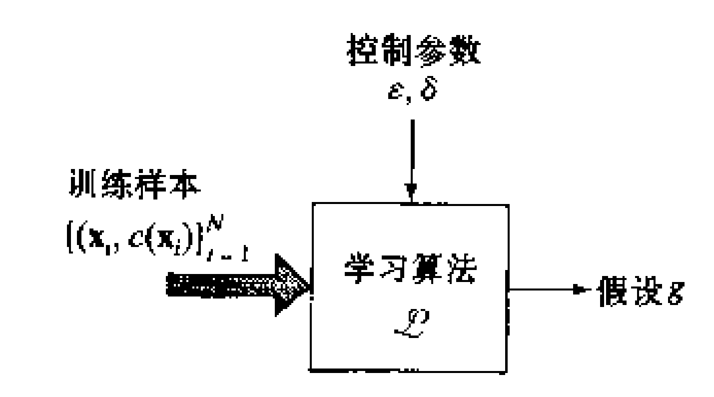


图2-26 PAC学习模型的框图

在此背景下我们现在可以将PAC学习模型正式地陈述如下(Valiant, 1984; Kearns and Vazirani, 1994; Vidyasagar, 1997)：

令$\mathscr{L}$为环境$\mathscr{X}$上的一个概念类。我们称概念类$\mathscr{L}$是PAC可学习的，如果存在一个算法$\mathscr{L}$具有如下性质：对于每一个目标概念$c\in\mathscr{L}$，对$\mathscr{X}$上的每个概率分布，以及对所有的$0<\varepsilon<1/2$和$0<\delta<1/2$，如果对学习算法$\mathscr{L}$提供训练例集$\mathscr{T}=\{(\mathbf{x}_i,c(\mathbf{x}_i))\}_{i=1}^N$以及参数$\varepsilon$和$\delta$，那么学习算法$\mathscr{L}$至少以概率$1-\delta$输出一个误差$\nu_{\text{train}}\leqslant\varepsilon$的假设$g$。这个概率是针对从集合$\mathscr{T}$中在抽取的随机样本以及可能存在于学习算法$\mathscr{L}$中的任何内部随机性而取得。样本大小$N$必须大于$\varepsilon$和$\delta$的一个函数。

换言之，只要训练样本$\mathscr{T}$的大小N足够大，在神经网络已在那个数据集上训练过之后，103 很可能的情况是，由网络计算的输入－输出映射是"近似正确的"。注意，虽然存在对ε和δ的依赖，例的数目N并不一定依赖于目标概念c或者$\mathscr{X}$的基本概率分布。

样本复杂性

在PAC学习理论中，对实际意义有特别影响的问题是样本复杂性问题。这一问题的焦

点在于需要提供给学习算法多少随机例使得算法能够获得足够的信息来学习一个从概念类$\mathscr{L}$选择的未知目标概念 c，或者训练集$\mathscr{T}$的大小应该多大？

样本复杂性问题与 VC 维密切相关。然而，在继续讨论这一问题之前，我们需要定义一个相容概念的含义。令$\mathscr{T}=\{(\mathbf{x}_i,d_i)\}_{i=1}^{N}$为任意标签例集，其中每个$\mathbf{x}_i\in\mathscr{X}$，每个$d_i\in(0,1)$。令 c 为在环境$\mathscr{X}$之上的目标概念。那么，说概念 c 是与训练集$\mathscr{T}$相容的（或者等价地，说$\mathscr{T}$是与 c 相容的），如果对所有 $1\leqslant i\leqslant N$ 我们有 $c(\mathbf{x}_i)=d_i$（Kearns and Vazarini, 1994）。现在就 PAC 学习而言，关键不是一个神经网络能计算的输入－输出函数集的数量，而是网络的 VC 维。更精确地说，我们有分成两部分给出的一个重要结论（Blumer et al., 1989; Anthony and Biggs, 1992; Vidyasagar, 1997）：

考虑具有有限 VC 维 $h\geqslant 1$ 的神经网络。

- *任何用于这个神经网络的相容学习算法是 PAC 学习算法。*
- *存在常数 K，使得对于任何这样的算法，大小为*

$$N=\frac{K}{\varepsilon}\left(h\log\left(\frac{1}{\varepsilon}\right)+\log\left(\frac{1}{\delta}\right)\right) \tag{2.107}$$

的训练集是足够的，其中 ε 为误差参数和 δ 为置信参数。

这个结果的普遍性是可喜的；它可应用于监督学习过程而不管使用什么类型的学习算法和用于生成标签例的基本概率分布。正是这一结论的高度一般性使其成为神经网络研究文献中受到广泛研究的主题。将基于 VC 维测量的界限预测的结果与试验结果比较，揭示了很大的差异[16]。在一定意义上这并不奇怪，因为这种差异仅仅是理论测量的*自由与分布无关的*、*最坏情况*的本质反映，并且在普通情况下我们总可以做得更好。

计算复杂性

在 PAC 学习中，需要着重关注的另一个问题是计算复杂性问题。这一问题涉及学习算法的计算有效性。更确切地说，*计算复杂性*是在给定具有有限大小 N 的分类标签样例时，涉及训练神经网络（学习机器）所需的最坏情况下的“运行时间”。 [104]

在实际情形中，算法的运行时间自然依赖于基本运算执行的速度。但是，从理论的角度看，目的是找到不依赖于计算所用设备的运行时间的定义。基于这个目的，运行时间以至计算复杂性通常从执行计算所需的操作（加法、乘法和存储）次数进行测量。

在估算学习算法的计算复杂性时，我们想知道它随样例大小 m（即被训练的神经网络的输入层的大小）是如何变化的。为了使算法在这种环境下在计算上是高效的，运行时间应该对于某一固定整数 $r\geqslant 1$ 为 $O(m^r)$。在这种情况下，说运行时间是随 m 多项式增长的，算法本身被称为一个*多项式时间算法*。由一个多项式时间算法执行的学习任务通常被认为是“容易的”（Anthony and Biggs, 1992）。

需要注意的另一参数是误差参数 ε。虽然在样本复杂性的情形中参数 ε 是固定而随意的，在估算学习算法的计算复杂性时我们想知道它随 ε 如何变化。直观上，我们预料当 ε 减小时研究中的学习任务会变得更困难。于是得出必须对算法要产生一个可能近似正确输出所花的时间施加一个条件。为了使计算是高效的，适当的条件是运行时间为 $1/\varepsilon$ 多项式的。

将这些考虑综合在一起，我们可以对计算复杂性作出如下形式化的陈述（Anthony and

Biggs，1992)：

一个学习算法关于误差参数ε、样例大小 m 和训练集的大小 N 是计算有效的，如果它的运行时间是关于 N 多项式的，并且对于PAC学习如果存在足够大的 $N_0(\delta,\varepsilon)$ 是关于 m 和 ε^{-1} 多项式的。

2.16 小结和讨论

在本章中，我们从神经网络的角度讨论了与学习过程的许多方面相关的一些重要问题。这样一来，就为本书余下部分中很多内容打下了基础。五个学习规则，即*误差－修正学习*、*基于记忆的学习*、*Hebb学习*、*竞争学习*和*Boltzmann学习*，是神经网络设计的基础。这些算法中一些需要使用一个教师，另一些则不需要。重要一点是这些规则使我们在能力和普遍性上都能超出线性自适应过滤器的范围。

在研究监督学习时，一个重要的条件是"教师"，它能够在误差－修正学习中发生错误时为网络输出提供精确的修正；或者像Boltzmann学习那样将网络自由运行的输入和输出单元"钳制"到环境。这两种模型在生物组织中都是不可能的。生物组织既没有用于反向传播误差修正(在多层前馈网络中)的精确的双向的神经连接，也不会有强制接受外部行为的神经方式。然而，正如第3章和第7章所展示的那样，监督学习已经确立了它在人工神经网络设计中作为一种有力范例的地位。

相反，自组织(无监督)学习规则(比如Hebb学习和竞争学习)是受神经生物学的思想启发的。但是，为了提高我们对自组织学习的理解，也需要参看Shannon的*信息论*以获得相关思想。这里我们应提到Linsker(1988a,b)的*最大互信息*(maximum mutual information，Infomax)*原则*，如同在通信信道中的信息传输那样，它为自组织神经网络中的信息处理提供了数学的形式化手段。Infomax原则及其变形在第10章讨论。

如果不提到*达尔文选择学习模型*(Edelman，1987；Reeke et al.，1990)，对学习方法的讨论将是不完全的。选择在进化和发展的应用中都是一个强有力的生物学原则。它居于已经透彻了解的生物认知系统即免疫系统的核心(Edelman，1973)。达尔文选择学习模型基于神经团选择理论。它预先假定，在每个动物生命期中脑神经系统以一种与进化中的自然选择性质类似的选择方式运作。根据这一理论，神经系统的基本操作单元不是单独的神经元，而是强连接的神经元的局部团。神经网络在一个团中的成员资格通过神经元的突触权值的改变而变化。神经元间的局部竞争和合作对形成网络中的局部顺序显然是必需的。一组神经团称为*指令系统*(repertoire)。一个指令系统的组由于神经生长的随机性质而对重叠但相似的模式有最好的响应。一个或更多的神经团响应每个输入模式，从而保证了对可能很重要但又出乎意料的输入模式有某种响应。达尔文选择学习与在神经网络设计中通常使用的学习算法的不同之处在于它假设设计了很多子网络，并且只有那些有期望响应的子网络才在训练过程中被选择。

我们以对学习的统计和概率方面的某些评述来结束这里的讨论。VC维已经成为统计学习理论中的核心参数。它对结构风险最小化和学习的可能近似正确(PAC)模型都是基本的。VC维是将在第6章讨论的所谓支持向量机基本理论的组成部分。在第7章，我们讨论一类基于推举(boosting)的委员会(committee)机，其理论植根于PAC学习。

当我们继续本书余下的部分时，会有很多情况和充分的理由来回顾本章中所提供的关于

学习过程的基础的内容。

注释和参考文献

[1] “算法”(algorithm)一词是从波斯数学家 Mohammed al-Kowârisimi 的名字而来，他生活在 9 世纪并且被认为发展了用于普通十进数的加、减、乘、除的分步规则。当他的名字用拉丁文书写时就变成了 Algorismus，Algorithm 就是这样衍生出来的(Harel,1987)。 106

[2] 大量文献包含了最近邻规则，参看 Dasarathy(1991)编辑的论文集，这本书包含了 Fix and Hodges(1951)的开创性工作以及许多其他关于最近邻模式分类技术的许多重要文章。

[3] 关于 Hebb 突触的详述，包括历史评述，参考 Brown et al.(1990)及 Frégnac 和 Schulz (1994)。另外的综述材料可参考 Constantine-Paton et al.(1990)。

[4] **长期电位—Hebb 突触的生理学证据**

Hebb(1949)为我们提供了考虑突触记忆机制的方法，但是近四分之一世纪过去后他的建议才获得实验证据的支持。1973 年，Bliss 和 Lomo 发表文章描述了在脑中称之为海马区中的激活导致突触改变的一种方式。他们对进入这个结构的主通道应用电刺激的冲击，同时记录引起突触的反应。当他们确信获得反应生物形态学的稳定基线特征时，他们应用简短的高频冲击训练。而当他们总结测试冲击的应用时，他们发现响应的振幅要大得多。记忆研究人员最感兴趣的是发现这种效果可以持续很长时间，他们称这种现象为长期电位(long-term potentiation,LTP)。

现在每年有几百篇关于 LTP 现象的论文发表，我们知道许多它的固有机制。例如，我们知道电位作用被限定在激活通路上。我们也知道 LTP 表现出许多联想性质。所谓联想性质我们是指同时活跃通路间的相互作用。在特别情况下，若一个正常情况下不会导致 LTP 效果的弱输入与一个强输入配对时，则弱输入被充电。这之所以被称为联想性质是因为它和学习系统的联想性质相类似。例如，在 Pavlov 条件反射试验中，一个神经(弱)听觉刺激和一个强(食物)刺激配对；这种配对产生条件反射的一种形式，对听觉刺激分泌唾液。

在这个领域的许多试验工作集中在 LTP 的联想性质。支持 LTP 的许多突触利用谷氨酸作为神经传导器。但是，实际上在后突触神经元中有许多不同的受纳器响应谷氨酸。所有这些受纳器有不同的性质，但我们仅考虑其中的两种性质。主要的突触响应是由 AMPA 受纳器的激活导致的(这些受纳器的名称是根据它们响应最强烈的药物的名称而来的，但它们都是谷氨酸受纳器)。当在一个 LTP 实验中记录一个响应时，它基本上是由于 AMPA 受纳器的激活的性质。在突触激活后，释放谷氨酸且和后突触膜的受纳器绑定。AMPA 受纳器的离子通道部分张开，导致作为突触基本响应的电流。

第二种类型的谷氨酸受纳器，即 NMDA 受纳器，有一些有趣的性质。和 NMDA 受纳器绑定的谷氨酸不足以开启相关的离子通道，通道保持关闭直到突触活跃(包括 AMPA 受纳器)产生足够大的电压差。因此，AMPA 受纳器为化学依赖的，而 NMDA 受纳器同时是化学依赖和电压依赖的。我们需另外的信息看清这个差异的重要性。和 AMPA 受纳器相关联的离子通道和纳离子的运动(它产生突触电流)联系。和 NMDA 受纳器相关联的离子通道允许钙进入细胞。虽然钙的运动也会影响膜电流，但其主要作用是作为触发信号，触发一连串的事件，导致和 AMPA 受纳器相关联的响应强度的持 107

续增加。

现在我们有关于Hebb突触的机制。NMDA受纳器要求前突触活跃(释放谷氨酸)和后突触活跃。这种情况怎样才能正常发生？保持足够强的输入就可以了。因而当我们将一个弱输入和一个强输入配对，弱输入释放它的谷氨酸，而强输入保证有足够强的电压差激活和弱突触相连接的NMPA受纳器。

虽然Hebb最初的建议仅限于单向学习规则，但如果利用*双向学习规则*，则可以认为神经网络更具有灵活性，突触权值既可以增加又可以减少是其优势。令人放心的是知道也有实验证据支持突触衰减机制。如果弱输入的激活不伴随强输入的激活，突触权值常常被减弱。这在突触系统的低频激活的响应中最为常见，这种现象称之为长期*衰减*(long-term depression，LTD)。也有一些证据表明称之为*奇异突触衰减*的作用。LTP限制为激活输入的衰减，而奇异突触衰减则为非激活输入。

[5] 竞争学习的思想可追溯到von der Malsburg(1973)的关于条纹皮质的方向敏感神经细胞的自组织，Fukushima(1975)的以*神经认知机*著名的自组织多层神经网络，Willshaw and von der Malsburg(1976)的自组织模型神经连接结构，以及Grossberg(1972，1976a，b)的自适应模式分类等的早期工作。并且有重要的证据表明竞争学习在脑组织映射结构中起着关键作用(Durbin et al.，1989)，最近Ambros-Ingerson et al.(1990)的实验工作提供竞争学习的进一步生理学上的证据。

[6] 如图2-4所示，利用侧抑制在神经生物系统很流行。大多数感觉组织，即眼球的视网膜，耳蜗及皮肤的触觉神经，都以这样一种方式组织，对任何给定位置的刺激都在周围神经元中产生抑制(Arbib，1989；Fischler and Firschein，1987)。在人类感知中，侧抑制表现在一种称之为*马赫带*(Mach band)的现象中，马赫带是根据物理学家Ernest Mach(1865)的名字来命名的。例如，如果我们看一张一半黑一半白的纸，即使它们有同样的密度，我们将会在白的部分看到比白更白的平行于边界的一个带，在黑的部分看见比黑更黑的平行于边界的一个带。马赫带不是物理上出现的，而是视觉上的幻觉，代表由侧抑制的差异动作引起的过投射或欠投射。

[7] John von Neumann深刻认识到统计热力学在研究计算机中的重要性。1949年他在Illinois大学所作的关于《复杂自动机的理论和组织》的五个报告的第三个中很好地说明了这一点。在他关于《信息的统计理论》的第三次讲演中，von Neumann指出：热力学概念也许将进入新的信息理论。有一些强烈的迹象显示信息类似于熵，并且熵的退化过程和信息处理中的退化过程是平行的。假如没有它运行的环境的统计特征，你是不能定义一个自动机的功能或效率的，正如在利用表征热力学环境的统计特征时一样。自动机环境的统计变量当然比标准热力学的温度变量复杂，但它们在特征上相似。

108

[8] 看来术语“增强式学习”是由Minsky(1961)在他的早期人工智能研究中创造的，然后由Waltz and Fu(1965)在控制论中独立提出。但是“增强式”的基本思想在心理学的动物学习实验研究中已出现(Hampson，1990)。在这个背景下，由Thorndike的下述经典*效果律*可以表明这一点(Thorndike，1911，p244)：

对于同一情况作出的几种不同响应，只有那些伴随或接近动物满足的或其他等同的东西才有可能和该情况更加紧密的联系，这样当它重新发生时，它们将更有可能发生；其他的那些伴随或接近使动物不舒服的或其他等同的东西，与那种情形的联系会

减弱，这样当它发生时，它们发生的可能减少。满足或不舒服的程度越大，联系带的增强或减弱的程度就越强。

虽然，不能说这个原理提供了一个生物行为的完整模型，但它的简单性和普通意义的方法使之成为增强式学习的传统方法中的一个有影响的学习规则。

[9] 设备输出是典型物理变量。为控制设备，我们需要清楚知道这个变量的值，即我们必须度量设备输出。用于度量一个物理变量的系统称为*感知器*，因而更准确地说，图2-13的方框图在它的反馈路径中应包括一个感知器。我们省略了感知器，暗示它的转移函数假定为单位的。

[10] "鸡尾酒会现象"指人类在噪声环境中挑选和跟踪听觉输入源的显著能力(Cherry,1953; Cherry and Taylor,1954)。这种能力表现在听觉系统所完成三种过程的组合中：

- *分割* 输入听觉信号被分割到单个频道，每个频道提供关于听者环境的有意义的信息。在分割时听者利用的所有启示中，*空间位置*也许是最重要的(Moray,1959)。
- *注意* 这包括听者集中注意在一个频道而忽略其他不相关频道的能力(Cherry,1953)。
- *转换* 第三个过程涉及从一个频道转换到另一个频道的能力，它也许通过"开启"输入听觉信号以由顶向下的方式调节(Wood and Cowan,1995)。

由这些观点可导出的结论是输入听觉信号所完成的处理确实是时空类型的。

[11] 设计最优线性滤波器问题提供了线性自适应滤波器的理论框架，这个问题首先由Kolmogorov(1942)提出并且不久后由Wiener(1949)独立解决。

另一方面，最优非线性滤波问题的形式解在数学上是不能解的。但是在50年代，Zadeh(1953)，Wiener及其合作者(Wiener,1958)作了大量出色的工作，而其他人对澄清问题的性质作了许多工作。 109

1954年Gabor是第一个认识到非线性自适应滤波器思想的人，并且随后在他的合作者帮助下建立了这种滤波器(Gabor et al.,1960)。基本上Gabor提出了绕过非线性自适应滤波数学困难的捷径，通过学习优化它的响应构造滤波器。滤波器输出形式上可表示为

$$y(n) = \sum_{n=0}^{N} w_n x(n) + \sum_{n=0}^{N}\sum_{m=0}^{N} w_{n,m} x(n) x(m) + \cdots$$

其中$x(0), x(1), \cdots, x(N)$是滤波器输入的采样。(这个多项式现在称之为*Gabor-Kolmogorov多项式*或*Volterra级数*。)多项式的第一项表示线性滤波器，由一组系数$\{w_n\}$表征。第二项由一组二元系数$\{w_{n,m}\}$表征，是非线性的；这项包含滤波器输入的两个样本的乘积，依次类推可得高阶项。滤波器的系数由梯度下降调整使得极小化目标(期望)响应$d(N)$和实滤波器输出$y(N)$之差的均方值。

[12] 式(2.71)中的代价函数$L(d, F(\mathbf{x}, \mathbf{w}))$应用于标量$d$。当期望响应为向量$\mathbf{d}$时，逼近函数采用向量值形式$\mathbf{F}(\mathbf{x}, \mathbf{w})$。这时我们用平方欧几里德距离

$$L(\mathbf{d}, \mathbf{F}(\mathbf{x}, \mathbf{w})) = \| \mathbf{d} - \mathbf{F}(\mathbf{x}, \mathbf{w}) \|^2$$

作为损失函数。函数$\mathbf{F}(\cdot, \cdot)$为它的变元的向量值函数。

[13] 根据Burges(1998)，首先出现在Vapnik(1995)中的例2.3归功于E.Levin和J.S.Denker。

[14] 线性阈值单元(感知器)构成的前馈网络VC维数的上界由Baum and Haussler(1989)获

得。随后，Maass(1993)证明，对于这类网络，一个更小的下界也成立，其数量级为 $W\log W$。

sigmoidal 神经网络的 VC 维数的第一个上界是 Macintyre and Sontag(1993)推出的。随后 Koiran and Sontag(1996)回答了 Maass(1993)提出的公开问题：

"具有 sigmoid 激活函数 $\sigma(y) = 1/(1+e^{-y})$ 的模拟神经网络的 VC 维数是否以可变参数个数的多项式为界？"

Koiran 和 Sontag 在他们 1996 年文章中明确回答了这个问题，正如前面所述。

Karpinski and Macintyre(1997)也明确回答了这个问题。在这后一篇文章中利用基于微分拓扑的复杂方法证明了模式分类器的 sigmoid 神经网络的 VC 维数的一个上界为 $O(W^4)$。这个上界和 Koiran 和 Sontag(1996)导出的下界之间有较大的差距。Karpinski and Macintyre(1997)猜想他们的上界可以降低。

[15] Sauer 定理可陈述为(Sauer, 1972; Anthony and Biggs, 1992; Vidyasagar, 1997)：

令$\mathscr{F}$表示学习机器实现的二分总体，若 $\mathrm{VCdim}(\mathscr{F}) = h$，$h$ 有限且 $l \geq h \geq 1$，那么增长函数 $\Delta_{\mathscr{F}}(l)$ 的界为 $(el/h)^h$，其中 e 为自然对数的底。

[16] 在这个注释中我们给出文献中报导的样本复杂性和相关的泛化问题的四个重要研究的总结。

首先，Cohn and Tesauro(1992)对基于 VC 维数的样本复杂性界作为模式分类器设计工具的实际价值给出详细的实验研究。特别是，设计了检验神经网络泛化能力和 Vapnik 统计学习理论导出的与分布无关的最坏情况界之间的关系的试验。这里考虑的界是 Vapnik(1982)定义的

110

$$\nu_{gene} \geq O\left(\frac{h}{N}\log\left(\frac{h}{N}\right)\right) \tag{1}$$

其中 ν_{gene} 是泛化误差，h 是 VC 维数，N 是训练集的大小。Cohn 和 Tesauro 给出的结果表明平均泛化能力比式(1)预测的好得多。

其次，Holden and Niranjan(1995)扩展了 Cohn 和 Tesauro 早期的研究，解决了一个相似的问题。但有三个重要差别需要指出：

- 神经网络所做的所有实验都知道 VC 维数的精确结果或非常好的界；
- 特别考虑了所用的学习算法；
- 实验采用现实生活中的数据。

虽然报告的结果发现提供样本复杂性预测比早期理论提供的值有意义得多，但是仍由许多理论缺陷有待克服。

第三，Baum and Haussler(1989)报告了训练线性阈值神经元的单层前馈网络具有良好泛化能力所需的训练样本大小 N。假设训练集从任意概率分布函数选择，并且评价泛化性能的测试样本服从相同的分布，那么，根据 Baum 和 Haussler，如果满足以下两个条件，网络几乎肯定提供泛化：

(1)对训练集产生错误的次数小于 $\varepsilon/2$

(2)训练中所用的样本数 N 为

$$N \geq O\left(\frac{W}{\varepsilon}\log\left(\frac{W}{\varepsilon}\right)\right) \tag{2}$$

其中 W 为网络突触权值数目。式(2)提供了与分布无关的最坏情况下 N 的界。这里，所需训练样本的实际数目和式(2)计算的界之间又有一个巨大的差异。

最后，在模式分类任务中用大的神经网络时，我们经常发现利用比 Cohn and Tesauro(1992)报告的网络权值数目小得多的训练样本数目表现良好，Bartlett(1997)讨论这个问题。在 Bartlett 的文章中证明，对于那种神经网络具有良好泛化而突触权值不是特别大的任务，是由权值的大小而不是权值数目决定网络的泛化性能。

习题

学习规则

2.1 式(2.3)描述的增量规则和式(2.9)描述的 Hebb 规则代表两类不同学习方法。列出这两个规则相互区别的特征。 111

2.2 利用禁止从输出中抽取期望响应(目标值)，再用反-Hebb规则(Mitchison，1989)，可以实现误差修正学习规则。讨论误差学习的这种解释。

2.3 图 2-27 表示二维平面数据点集。一部分数据点集属于类$\mathscr{C}_1$ 而另一部分数据点集属于类$\mathscr{C}_2$。对该数据集构造应用最近邻规则产生的判定边界。

2.4 考虑一组人，把他们关于某主题的集体意见定义为每个成员各自意见的加权平均。假设在讨论过程中，成员的意见和集体意见趋向一致，则他的意见的权值增加，另一方面，如果成员总是不同意集体意见，那么他的意见的权值减小。这样加权形式等价于正反馈控制，它有在组内产生一致意见的效果(Linsker，1988a)。

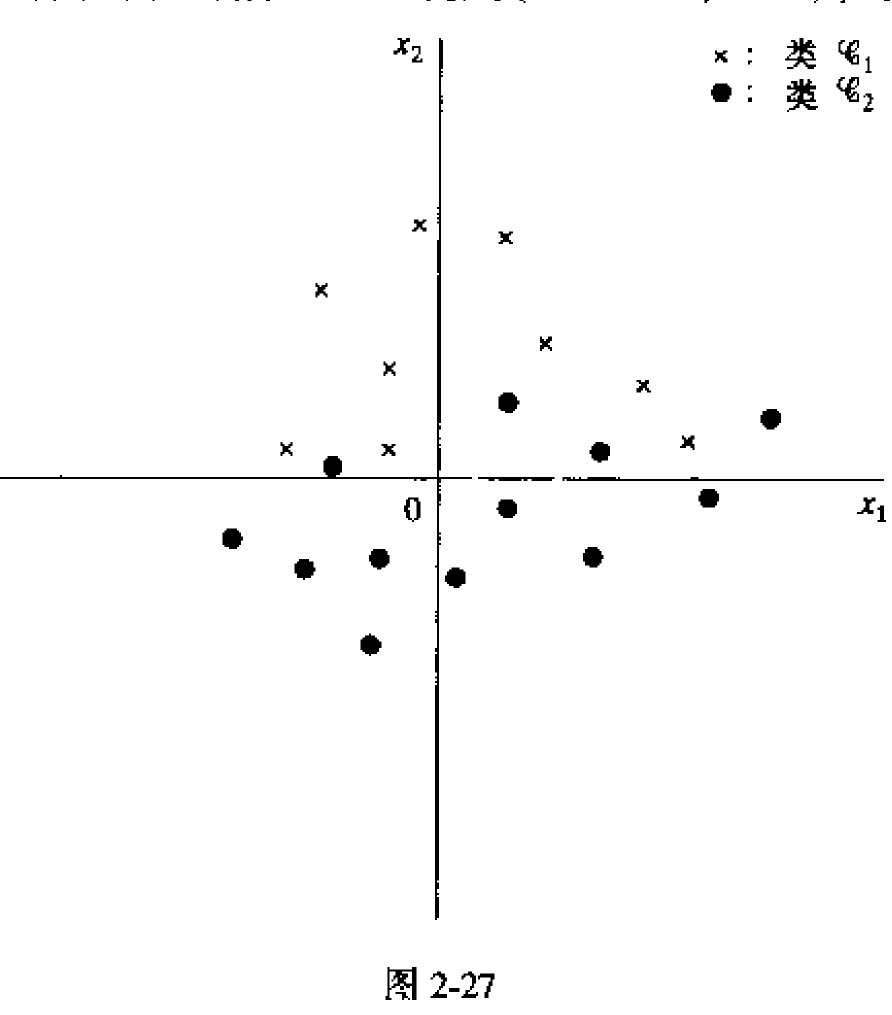


图 2-27

讨论所描述的情况和学习的 Hebb 假说的类似之处。

2.5 一个 Hebb 规则的广义形式可描述为：

$$\Delta w_{kj}(n) = \alpha F(y_k(n))G(x_j(n)) - \beta w_{kj}(n)F(y_k(n))$$

其中 $x_j(n)$和 $y_k(n)$为前突触和后突触信号；$F(\cdot)$和 $G(\cdot)$为它们各自变量的函数；$\Delta w_{kj}(n)$为在时刻 n 时突触权值 w_{kj}关于信号 $x_j(n)$和 $y_j(n)$的响应产生的改变量。寻找(a)平衡点和(b)这个规则定义的最大衰减。

2.6 一个幅度为 1 的输入信号重复应用于初值为 1 的突触连接。计算利用下面两个规则时突触权值的偏差：

(a)在式(2.9)中描述的 Hebb 规则的简单形式，假设学习率参数 $\eta = 0.1$。 112

(b)在式(2.10)中描述的协方差规则，假设前突触活动 $\bar{x} = 0$ 而后突触活动 $\bar{y} = 1.0$。

2.7 在式(2.9)中描述的 Hebb 突触涉及使用正反馈。验证这个陈述的正确性。

2.8 考虑式(2.10)中描述的关于自组织学习的协方差假说。假设遍历(即时间平均可替代总体平均)，证明在式(2.10)中的 $\Delta w_{kj}(n)$ 的期望值可表示为

$$E[\Delta w_{kj}] = \eta(E[y_k x_j] - \bar{y}\bar{x})$$

你怎样解释这个结果。

2.9 根据 Linsker(1986)，学习的 Hebb 假说可以用公式

$$\Delta w_{ki} = \eta(y_k - y_o)(x_i - x_o) + a_1$$

表示，其中 x_j 和 y_k 分别为前突触和后突触信号，a_1，η，x_0，y_0 都是常数。假设神经元 k 是线性的，由

$$y_k = \sum_j w_{kj} x_j + a_2$$

表示，其中 a_2 为另一常数。假设所有输入信号的概率分布相同，即 $E[x_i] = E[x_j] = \mu$。令矩阵 **C** 表示为输入信号的协方差矩阵，它的第 ij 个元素定义为

$$c_{ij} = E[(x_i - \mu)(x_j - \mu)]$$

试确定 $E[\Delta w_{ki}]$。

2.10 给出图 2-28 网络中神经元 j 的输出 y_j 的表达式。你可应用下列量：

x_i = 第 i 个输出信号

w_{ji} = 从输入 i 到神经元 j 的突触权值

c_{kj} = 从神经元 k 到神经元 j 的侧向连接的权值

v_j = 神经元 j 的诱导局部域

$y_j = \varphi(v_j)$

神经元 j 成为获胜神经元应该满足什么条件?

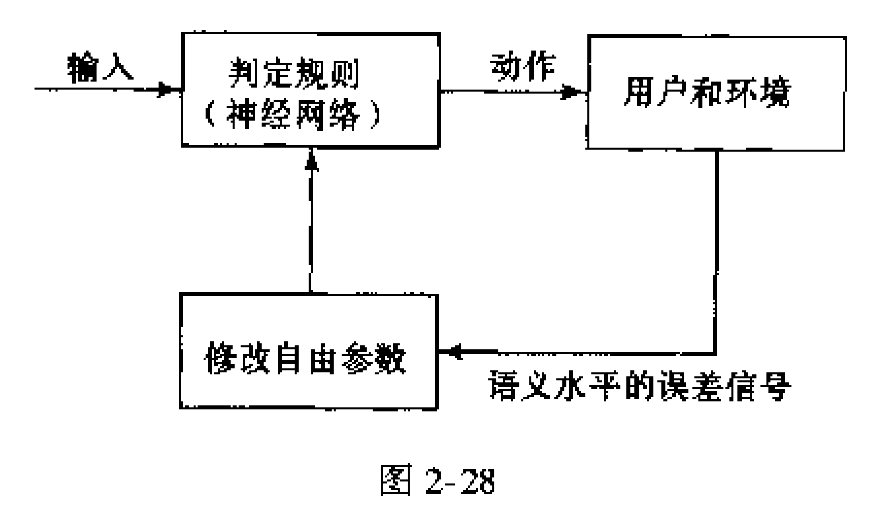


图 2-28

2.11 假设每个输出神经元包括自反馈，重复问题 2.10。

2.12 侧抑制的连接模式，即"近激励而远抑制"，可以用两个 Gauss 曲线的差建模。这两条曲线有相同的面积，但是，用于激励的正曲线比用于抑制的负曲线有较高和较窄的峰。也就是可以把连接模式表示为：

$$W(x) = \frac{1}{\sqrt{2\pi}\sigma_e} e^{-x^2/2\sigma_e^2} - \frac{1}{\sqrt{2\pi}\sigma_i} e^{-x^2/2\sigma_i^2}$$

其中 x 是到侧抑制神经元的距离。利用模式 $W(x)$ 扫描一个页面，一半是白的一半是黑的，两半之间的边界垂直于 x 轴。

113 画出当 $\sigma_e = 5$，$\sigma_i = 8$ 和 $\sigma_e = 1$，$\sigma_i = 2$ 时这个扫描过程的输出。

学习范例

2.13 图 2-28 给出自适应语言获得系统的方框图(Gorin, 1992)。根据机器对输入刺激响应的适应程度的反馈，系统的神经网络部分的突触连接被增强或减弱。这个系统可看作增强式学习的例子。说明这个陈述合理性。

2.14 下例算法中，哪两个范例属于有教师学习和无教师学习?

(a)最近邻规则

(b) k - 最近邻规则

(c) Hebb 学习

(d) Boltzmann 学习规则

说明你的答案的理由。

2.15 无监督学习可以用在线或离线方式实现。讨论这两种可能方式的物理含义。

2.16 考虑学习机器面对象棋游戏结果(赢、输或平局)信任赋值的困难。在这个游戏背景下讨论时间信任赋值和结构信任赋值的概念。

2.17 可以把一个监督学习任务看作增强式学习任务，其中把系统的实际响应和期望响应靠近的某种度量作为增强信号。讨论监督学习和增强式学习的这种关系。

2.18 考虑应用于相关矩阵记忆的关键模式的下述正交集：

$$\mathbf{x}_1 = [1,0,0,0]^T \qquad \mathbf{x}_2 = [0,1,0,0]^T \qquad \mathbf{x}_3 = [0,0,1,0]^T$$

相应的储存模式为

$$\mathbf{y}_1 = [5,1,0]^T \qquad \mathbf{y}_2 = [-2,1,6]^T \qquad \mathbf{y}_3 = [-2,4,3]^T$$

114

(a)计算记忆矩阵 $\mathbf{M}$。

(b)证明记忆完全联想。

2.19 再考虑问题 2.18 的相关矩阵记忆。应用于记忆的刺激是关键模式 $\mathbf{x}_1$ 的带噪声形式，表示为

$$\mathbf{x} = [0.8, -0.15, 0.15, -0.20]^T$$

(a)计算记忆响应 $\mathbf{y}$。

(b)证明响应 $\mathbf{y}$ 在欧几里德意义下和存储模式 $\mathbf{y}_1$ 最接近。

2.20 利用下例关键向量训练自联想记忆：

$$\mathbf{x}_1 = \frac{1}{4}[-2, -3, \sqrt{3}]^T \qquad \mathbf{x}_2 = \frac{1}{4}[2, -2, -\sqrt{8}]^T \qquad \mathbf{x}_3 = \frac{1}{4}[3, -1, \sqrt{6}]^T$$

(a)计算这些向量之间的夹角。它们相互之间离正交性有多近?

(b)利用推广的 Hebb 规则(即外积规则)，计算网络的记忆矩阵。考查自联想和完全记忆联想有多近。

(c)把关键向量 $\mathbf{x}_1$ 的伪装形式即

$$\mathbf{x} = [0, -3, \sqrt{3}]^T$$

应用于记忆。计算记忆的响应，将结果和期望响应 $\mathbf{x}_1$ 比较。

自适应

2.21 图 2-29 表示一个自适应系统的方框图。预测模型的输入信号定义为过程的过去值，表示为

$$\mathbf{x}(n-1) = [x(n-1), x(n-2), \cdots, x(n-m)]^T$$

模型输出 $\hat{x}(n)$ 表示对过程现在值 $x(n)$ 的估计。比较器计算误差信号

$$e(n) = x(n) - \hat{x}(n)$$

它接着用于修正模型的可调参数。它也提供转

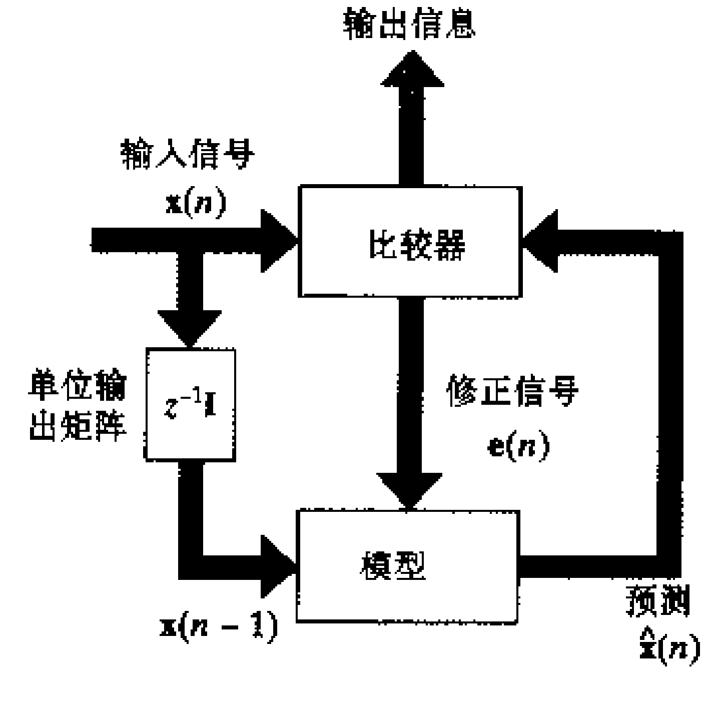


图 2-29

115

移到神经处理下一级的输出信号用于解释。在一级接一级的基础上重复这个操作，系统处理的信息逐步提高质量(Mead,1990)。

写出图2-29中描述的下一级信号处理的细节。

统计学习理论

2.22　根据从式(2.61)导出(2.62)的相似过程，导出式(2.66)定义的总体平均函数 $L_{av}(f(\mathbf{x}), F(\mathbf{x},\mathscr{T}))$ 的公式。

2.23　在这个问题中我们希望计算具有和平面上的坐标轴重合的矩形区域的VC维数。证明这个概念的VC维数为4。你可以通过下列方式完成证明：

(a)平面上的四个点，以及有边与一个坐标轴重合的矩形能够实现的二分；

(b)平面上四个点，以及有边与一个坐标轴重合的矩形不能够实现的二分；

(c)平面上五个点，以及有边与一个坐标轴重合的矩形也能够实现的二分。

2.24　考虑线性二值模式分类器，它的输入向量 $\mathbf{x}$ 有 m 维，向量 $\mathbf{x}$ 的第一个分量为常数1从而分类器相应的权值为偏置。分类器关于输入空间的VC维数是多少？

2.25　不等式(2.97)定义一致收敛速度的一个界，它是经验风险最小化原则的基础。

(a)假设不等式(2.97)成立，验证式(2.98)的正确性。

(b)导出定义置信区间 ε_1 的等式(2.99)。

2.26　继续例2.3，证明图2-30中的四个平均分布的点不能被单参数指示函数族 $f(x, a)$，$a \in \mathbb{R}$ 分散。

116

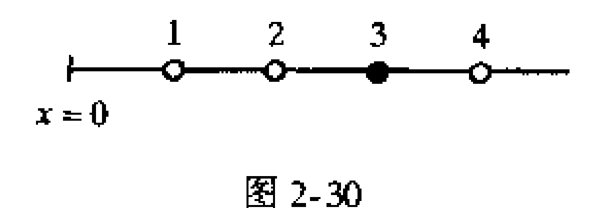


图2-30

2.27　在非线性回归环境下讨论偏置－方差困境和结构风险最小化的关系。

2.28　(a)由具有sigmoid函数的神经元组成的多层前馈网络的训练算法是PAC可学习的。验证这个陈述的正确性。

(b)由具有阈值激活函数的神经元组成的任意网络你能作出类似的陈述吗？证明你的答案的正确性。

第3章 单层感知器

3.1 简介

在神经网络的形成阶段(1943-1958)，一些研究者作出了开拓性的贡献：

- McCulloch and Pitts(1943)引入神经网络的概念作为计算的工具。
- Hebb(1949)提出自组织学习的第一个规则。
- Rosenblatt(1958)提出感知器作为有教师学习(即监督学习)的第一个模型。

McCulloch-Pitts关于神经网络的论文所造成的重要影响在第1章中已经得到了充分阐述。Hebb学习的概念某种程度上在第2章中也得到了讨论。在本章中我们将讨论Rosenblatt的*感知器*。

感知器是用于*线性可分*模式(即模式分别位于超平面所分隔开的两边)分类的最简单的神经网络模型。基本上它由一个具有可调突触权值和偏置的神经元组成。用来调整这个神经网络中自由参数的算法最早出现在Rosenblatt(1958,1962)提出的用于其脑感知模型的一个学习过程中[1]。事实上，如果用来训练感知器的模式(向量)取自两个线性可分的类，Rosenblatt证明了感知器算法是收敛的，而且由超平面构成的决策面位于两类之间。算法收敛性的证明被称为*感知器收敛定理*。建立在一个神经元上的感知器的模式分类被限制为只能完成两类(假设)的模式分类。通过扩展感知器的输出层可以使感知器包括不止一个神经元，相应地我们可以进行多于两类的分类。但是，只有这些类是线性可分时感知器才能正常工作。重要的一点在于仅关心作为模式分类器的感知器的基本理论，我们只需考虑单个神经元的情况。有关多个神经元的理论推广是很平常的。

单个神经元也构成一个*自适应滤波器*的基础，自适应滤波器是不断发展的信号处理主题的一个基本功能模块。自适应滤波器的发展很大程度上要归功于Widrow and Hoff(1960)有关*最小均方*(least mean square,LMS)*算法*(也被称为delta规则)的经典论文。LMS算法虽然实现很简单，但在应用中有很高的效率。事实上，它在线性自适应滤波中起着关键作用，线性指的是神经元在线性模型下运行。自适应滤波器在天线、通信系统、控制系统、雷达、声纳、地震学和生物医学工程等很多领域都有应用(Widrow and Stearns,1985;Haykin,1996)。

LMS算法和感知器本质上是相关的。因此我们把它们放在同一章里来学习是适宜的。

本章的组织

本章分为两部分。第一部分包括3.2节至3.7节，处理线性自适应滤波器和LMS算法；第二部分包括3.8至3.10节，处理Rosenblatt的感知器。从表示的观点看，我们发现先讨论线性自适应滤波器再讨论Rosenblatt感知器较为方便，这和它们在历史上出现的顺序相反。

在3.2节讨论自适应滤波问题，接着在3.3节回顾三种无约束最优化技巧：最速下降法、Newton法和Gauss-Newton法，它们都是与自适应滤波器研究有关的。3.4节讨论线性最小二乘滤波器，它随着数据长度的增加渐近趋于Wiener滤波器。Wiener滤波器为线性自适

应滤波器在平稳环境下的运行性能提供一个理想的框架。在3.5节描述LMS算法，包括它的优点和局限性。在3.6节探讨通常用来评价自适应滤波器性能的学习曲线的思想。3.7节讨论LMS算法的退火时间表。

随后转向到Rosenblatt的感知器，3.8节提供一些与其运行有关的基本考虑。3.9节描述应用于线性可分类别模式分类的感知器突触权值向量的调整算法，并验证此算法的收敛性。在3.10节考虑感知器和Gauss环境下Bayes分类器的关系。

本章以3.11节的总结和讨论作为结束。

3.2 自适应滤波问题

考虑一个动态系统，其数学特征未知。我们已知的是此系统在离散时间内以固定速率产生的一系列标定的输入－输出数据。具体地，当一个 m 维的刺激 $\mathbf{x}(i)$ 通过此系统的 m 个输入节点，系统产生一个标量输出 $d(i)$ 作为响应，如图3-1a所示，其中 $i=1,2,\cdots,n,\cdots$。此系统的外部行为由数据集

$$\mathscr{T}:\{\mathbf{x}(i),d(i);i=1,2,\cdots,n,\cdots\} \quad (3.1)$$

描述，其中

$$\mathbf{x}(i)=[x_1(i),x_2(i),\cdots,x_m(i)]^T$$

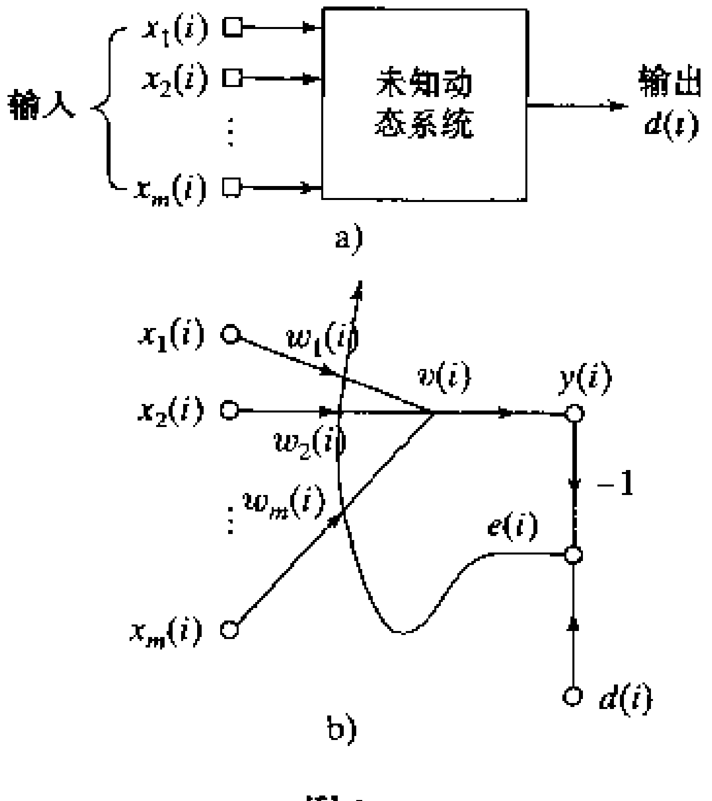


图3-1 a)未知动态系统 b)系统自适应模型的信号流图

$\mathscr{T}$ 中的样本根据一个未知概率法则是同分布的。输入向量 $\mathbf{x}(i)$ 的维数称为输入空间的维数或简称为维数(dimensionality)。

刺激 $\mathbf{x}(i)$ 能够以两种根本不同的方式之一出现，一种是空间的和另一种是时间的：

- $\mathbf{x}(i)$ 的 m 个元素代表空间中的不同点，在这种情况下我们称 $\mathbf{x}(i)$ 为数据的瞬像(snapshot)。
- $\mathbf{x}(i)$ 的 m 个元素代表在时间上均匀分布的某个刺激的现在和 $m-1$ 个过去的值组成的集合。

我们面对的问题是如何通过建立一个简单线性神经元来设计未知动态系统的一个多输入－单输出模型。这个神经元模型是在一个算法的影响下运行的，此算法控制对神经元的突触权值的必要调整，同时记住以下要点：

- 此算法从任意设定的一个神经元突触权值开始。
- 为响应系统行为的统计变化，突触权值调整是建立在连续基础上的(即把时间加进算法中)。
- 调整突触权值的计算在长度为一个采样周期的时间段里完成。

这样描述的神经元模型称为自适应滤波器(adaptive filter)。虽然在作为系统辨识的一个任务背景下给出的描述，但自适应滤波器的特征还是对很广的应用有足够的一般性。

图3-1b是一个自适应滤波器的示意图，它的运行由两个连续过程组成：

1. 过滤过程，涉及两个信号计算：

- 一个输出，记为 $y(i)$，它被产生以响应刺激向量 $\mathbf{x}(i)$ 的 m 个元素，即 $x_1(i)$，$x_2(i),\cdots$，$x_m(i)$。

- 一个误差信号，记为 $e(i)$，它是通过比较输出 $y(i)$和未知系统的相应输出 $d(i)$。事实上，$d(i)$作为一个期望响应信号或者目标信号。

2. 自适应过程，包括根据误差 $e(i)$对神经元突触权值的自动调整。

从而，这两个共同运作过程的组合构成一个围绕神经元运作的反馈环。

因为神经元是线性的，输出 $y(i)$恰为诱导局部域 $v(i)$；即

$$y(i) = v(i) = \sum_{k=1}^{m} w_k(i)x_k(i) \tag{3.2}$$

其中 $w_1(i)$，$w_2(i)$，…，$w_m(i)$表示在时刻 i 神经元的 m 个突触权值。利用矩阵形式我们可以表示 $y(i)$为向量 $\mathbf{x}(i)$和 $\mathbf{w}(i)$的内积形式如下：

$$y(i) = \mathbf{x}^T(i)\mathbf{w}(i) \tag{3.3}$$

这里 $$\mathbf{w}(i) = [w_1(i), w_2(i), \cdots, w_m(i)]^T$$

注意这个突触权值的记号已被简化，不包括附加的标识神经元的下标，因为我们只考虑单个神经元。这种考虑贯穿整个一章。神经元的输出 $y(i)$要与未知系统在时刻 i 的相应输出 $d(i)$作比较。通常，$y(i)$与 $d(i)$不等；因此它们的比较结果得到了误差信号：

$$e(i) = d(i) - y(i) \tag{3.4}$$

误差信号 $e(i)$用来对神经元突触权值调整进行控制的方式是由用于导出自适应滤波算法的代价函数决定的。这个问题与最优化紧密相关。因此回顾一下无约束最优化方法是适宜的。这些材料不仅可以应用在线性自适应滤波器上，还可以应用在一般神经网络上。 120

3.3 无约束最优化技术

考虑代价函数$\mathscr{E}(\mathbf{w})$，它是一个以未知权值(参数)向量 $\mathbf{w}$ 的连续可微函数。函数$\mathscr{E}(\mathbf{w})$映射 $\mathbf{w}$ 的元素为实数。它是一种度量，用来选择自适应滤波算法的权值(参数)向量 $\mathbf{w}$ 使得它以最优方式运行。我们想找到一个最优解 $\mathbf{w}^*$ 满足条件

$$\mathscr{E}(\mathbf{w}^*) \leqslant \mathscr{E}(\mathbf{w}) \tag{3.5}$$

也就是说，需要解决一个无约束的优化问题，即

选择适当的权值向量 $\mathbf{w}$ 最小化代价函数$\mathscr{E}(\mathbf{w})$ (3.6)

最优性的必要条件是

$$\nabla \mathscr{E}(\mathbf{w}^*) = \mathbf{0} \tag{3.7}$$

这里∇是梯度算子

$$\nabla = \left[\frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, \cdots, \frac{\partial}{\partial w_m}\right]^T \tag{3.8}$$

同时$\nabla \mathscr{E}(\mathbf{w})$是代价函数的梯度向量

$$\nabla \mathscr{E}(\mathbf{w}) = \left[\frac{\partial \mathscr{E}}{\partial w_1}, \frac{\partial \mathscr{E}}{\partial w_2}, \cdots, \frac{\partial \mathscr{E}}{\partial w_m}\right]^T \tag{3.9}$$

一种特别适合自适应滤波器设计的无约束最优化算法是以局部迭代下降思想为基础的：

以一个初始估计值 $\mathbf{w}(0)$开始，产生一系列权值向量 $\mathbf{w}(1)$，$\mathbf{w}(2)$，…，使得代价函数$\mathscr{E}(\mathbf{w})$在算法的每次迭代中要有下降，即

$$\mathscr{E}(\mathbf{w}(n+1)) < \mathscr{E}(\mathbf{w}(n)) \tag{3.10}$$

这里 $\mathbf{w}(n)$是权值向量的旧值而 $\mathbf{w}(n+1)$是它的更新值。

我们希望算法最终收敛到最优解 $\mathbf{w}^*$。我们说"希望"是因为除非采取特别的预防措施，算法有可能发散(即变得不稳定)。

在这一节我们描述三种以迭代下降思想这种或那种形式为基础的无约束最优化方法(Bertsekas,1995a)。

最速下降法

在最速下降法中，对权值向量 $\mathbf{w}$ 的连续调整是在最速下降的方向进行的，也就是它是与梯度向量 $\nabla\mathscr{E}(\mathbf{w})$ 方向相反的。为了表示方便，记为

121

$$\mathbf{g} = \nabla\mathscr{E}(\mathbf{w}) \tag{3.11}$$

因此，最速下降法一般表示为

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta\mathbf{g}(n) \tag{3.12}$$

这里 η 是一个正常数，称为*步长*(stepsize)或*学习率参数*(learning-rate parameter)，$\mathbf{g}(n)$ 是在 $\mathbf{w}(n)$ 处的梯度向量值。在从迭代 n 到 $n+1$ 的过程中算法应用修正

$$\Delta\mathbf{w}(n) = \mathbf{w}(n+1) - \mathbf{w}(n) = -\eta\mathbf{g}(n) \tag{3.13}$$

式(3.13)实际上是第2章中描述过的误差修正公式的标准形式。

为了证明最速下降法的公式满足式(3.10)的迭代下降条件，我们用 $\mathbf{w}(n)$ 附近的一阶 Taylor 级数展开来逼近 $\mathscr{E}(\mathbf{w}(n+1))$，即

$$\mathscr{E}(\mathbf{w}(n+1)) \simeq \mathscr{E}(\mathbf{w}(n)) + \mathbf{g}^T(n)\Delta\mathbf{w}(n)$$

上式对较小的 η 是适用的。在这个近似关系代入式(3.13)得到

$$\mathscr{E}(\mathbf{w}(n+1)) \simeq \mathscr{E}(\mathbf{w}(n)) - \eta\mathbf{g}^T(n)\mathbf{g}(n) = \mathscr{E}(\mathbf{w}(n)) - \eta\|\mathbf{g}(n)\|^2$$

上式表明，对正的学习率参数 η 代价函数每次迭代都是下降的。但这里提供的推导是近似的，只有当学习率足够小时才是正确的。

最速下降法收敛到最优解 $\mathbf{w}^*$ 的速度是很慢的。此外，学习率参数 η 对收敛速度有重要影响：

- 当 η 较小时，算法的瞬时响应是平缓的(overdamped)，由于 $\mathbf{w}(n)$ 的轨迹是 $\mathscr{W}$ 平面的一个光滑曲线，如图 3-2a 所示
- 当 η 较大时，算法的瞬时响应是剧烈的(underdamped)，由于 $\mathbf{w}(n)$ 的轨迹是锯齿(振荡)形的，如图 3-2b 所示。
- 当 η 超过了某一临界值时，算法是不稳定的(即不收敛的)。

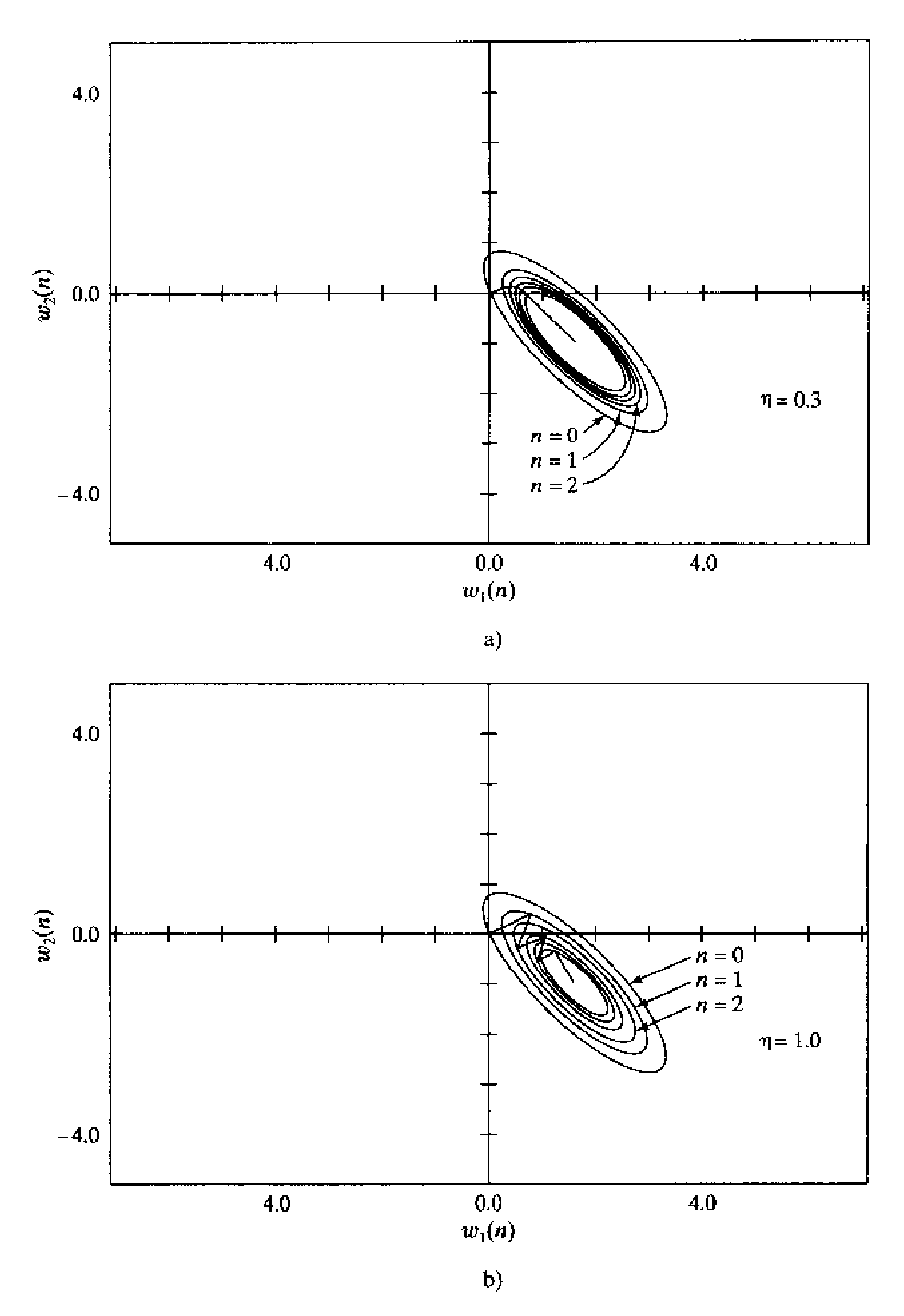
Newton 方法

Newton 方法的基本思想是最小化代价函数 $\mathscr{E}(\mathbf{w})$ 在当前点 $\mathbf{w}(n)$ 周围的二次近似值；最小化在算法的每次迭代中都要进行。特别，利用代价函数在点 $\mathbf{w}(n)$ 周围的二次 Taylor 级数展开式，我们得到

$$\Delta\mathscr{E}(\mathbf{w}(n)) = \mathscr{E}(\mathbf{w}(n+1)) - \mathscr{E}(\mathbf{w}(n)) \simeq \mathbf{g}^T(n)\Delta\mathbf{w}(n) + \frac{1}{2}\Delta\mathbf{w}^T(n)\mathbf{H}(n)\Delta\mathbf{w}(n) \tag{3.14}$$

122

和以前一样，$\mathbf{g}(n)$ 是代价函数 $\mathscr{E}(\mathbf{w})$ 在点 $\mathbf{w}(n)$ 处的 $m\times 1$ 梯度向量。矩阵 $\mathbf{H}(n)$ 是 $\mathscr{E}(\mathbf{w})$ 在 $\mathbf{w}(n)$ 的 m 行 m 列 Hessian 矩阵。$\mathscr{E}(\mathbf{w})$ 的 Hessian 矩阵定义为



123

图 3-2 最速下降法关于学习率参数的不同值在二维空间的轨迹：
a) $\eta=0.3$ b) $\eta=1.0$ 坐标 w_1 和 w_2 是权值向量 $\mathbf{w}$ 的元素

$$\mathbf{H}=\nabla^2\mathscr{E}(\mathbf{w})=\begin{bmatrix}\dfrac{\partial^2\mathscr{E}}{\partial w_1^2} & \dfrac{\partial^2\mathscr{E}}{\partial w_1\partial w_2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_1\partial w_m}\\ \dfrac{\partial^2\mathscr{E}}{\partial w_2\partial w_1} & \dfrac{\partial^2\mathscr{E}}{\partial w_2^2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_2\partial w_m}\\ \vdots & \vdots & & \vdots\\ \dfrac{\partial^2\mathscr{E}}{\partial w_m\partial w_1} & \dfrac{\partial^2\mathscr{E}}{\partial w_m\partial w_2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_m^2}\end{bmatrix} \tag{3.15}$$

式(3.15)需要代价函数$\mathscr{E}(\mathbf{w})$关于$\mathbf{w}$的元素二阶连续可微。对式(3.14)取$\Delta\mathbf{w}$微分[2]，当

$$\mathbf{g}(n)+\mathbf{H}(n)\Delta\mathbf{w}(n)=\mathbf{0}$$

时改变量$\Delta\mathscr{E}(\mathbf{w})$达到最小。解有关$\Delta\mathbf{w}(n)$的方程得到

$$\Delta\mathbf{w}(n)=-\mathbf{H}^{-1}(n)\mathbf{g}(n)$$

也就是

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\Delta\mathbf{w}(n)=\mathbf{w}(n)-\mathbf{H}^{-1}(n)\mathbf{g}(n) \tag{3.16}$$

这里$\mathbf{H}^{-1}(n)$是$\mathscr{E}(\mathbf{w})$的 Hessian 矩阵的逆。

一般来说，Newton 方法收敛得很快，而且不会出现最速下降法有时会出现的锯齿形情况。但是，应用 Newton 方法时，Hessian 矩阵必须对每个n都是正定矩阵[3]。不过，一般不能保证在算法的每次迭代中$\mathbf{H}(n)$都是正定矩阵。假如 Hessian 矩阵$\mathbf{H}(n)$不正定，修正 Newton 方法就有必要(Powell, 1987; Bertsekas, 1995a)。

Gauss-Newton 方法

Gauss-Newton 方法应用于这样一种代价函数，它表示为误差的平方和。令

124

$$\mathscr{E}(\mathbf{w})=\frac{1}{2}\sum_{i=1}^{n}e^2(i) \tag{3.17}$$

这里尺度因子 1/2 是为了简化以下的分析。此公式中的所有的误差项都是以权值向量$\mathbf{w}$为基础计算得来的，这里$\mathbf{w}$在遍及$1\leqslant i\leqslant n$的全部观察区间内固定。

误差信号$e(i)$是可调权值向量$\mathbf{w}$的函数。给定操作点$\mathbf{w}(n)$，我们通过以下方式来线性化$e(i)$对$\mathbf{w}$的依赖性：

$$e'(i,\mathbf{w})=e(i)+\left[\frac{\partial e(i)}{\partial \mathbf{w}}\right]^T_{\mathbf{w}=\mathbf{w}(n)}(\mathbf{w}-\mathbf{w}(n)),i=1,2,\cdots,n \tag{3.18}$$

用矩阵记号可写成等价的形式

$$\mathbf{e}'(n,\mathbf{w})=\mathbf{e}(n)+\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n)) \tag{3.19}$$

其中$\mathbf{e}(n)$是误差向量

$$\mathbf{e}(n)=[e(1),e(2),\cdots,e(n)]^T$$

$\mathbf{J}(n)$是$\mathbf{e}(n)$的$n\times m$ Jacobi 矩阵：

$$\mathbf{J}(n)=\begin{bmatrix}\frac{\partial e(1)}{\partial w_1} & \frac{\partial e(1)}{\partial w_2} & \cdots & \frac{\partial e(1)}{\partial w_m}\\ \frac{\partial e(2)}{\partial w_1} & \frac{\partial e(2)}{\partial w_2} & \cdots & \frac{\partial e(2)}{\partial w_m}\\ \vdots & \vdots & & \vdots\\ \frac{\partial e(n)}{\partial w_1} & \frac{\partial e(n)}{\partial w_2} & \cdots & \frac{\partial e(n)}{\partial w_m}\end{bmatrix}_{\mathbf{w}=\mathbf{w}(n)} \tag{3.20}$$

Jacobi 矩阵$\mathbf{J}(n)$是$m\times n$梯度矩阵$\nabla\mathbf{e}(n)$的转置，这里

$$\nabla\mathbf{e}(n)=[\nabla e(1),\nabla e(2),\cdots,\nabla e(n)]$$

更新的权值向量$\mathbf{w}(n+1)$定义为

$$\mathbf{w}(n+1)=\arg\min_{\mathbf{w}}\left\{\frac{1}{2}\|\mathbf{e}'(n,\mathbf{w})\|^2\right\} \tag{3.21}$$

利用等式(3.19)来估计$\mathbf{e}'(n,\mathbf{w})$的 Euclid 范数的平方，我们得到

$$\frac{1}{2}\|\mathbf{e}'(n,\mathbf{w})\|^2=\frac{1}{2}\|\mathbf{e}(n)\|^2+\mathbf{e}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n))$$
$$+\frac{1}{2}(\mathbf{w}-\mathbf{w}(n))^T\mathbf{J}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n))$$

因此，将以上表示方式对 w 求微分并设结果为零，我们得到 125

$$\mathbf{J}^T(n)\mathbf{e}(n)+\mathbf{J}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n))=\mathbf{0}$$

从这个方程中解出 w，考虑到式(3.21)我们可写为：

$$\mathbf{w}(n+1)=\mathbf{w}(n)-(\mathbf{J}^T(n)\mathbf{J}(n))^{-1}\mathbf{J}^T(n)\mathbf{e}(n) \tag{3.22}$$

上式描述 Gauss-Newton 方法的纯粹形式。

不像 Newton 方法必须知道代价函数$\mathscr{E}(n)$的 Hessian 矩阵，Gauss-Newton 方法只需要已知误差向量 $\mathbf{e}(n)$的 Jacobi 矩阵。但是，为了使 Gauss-Newton 迭代可计算，矩阵乘积 $\mathbf{J}^T(n)\mathbf{J}(n)$必须是非奇异的。

关于后一点，我们认识到 $\mathbf{J}^T(n)\mathbf{J}(n)$总是非负定的。为了保证它是非奇异的，Jacobi 矩阵 $\mathbf{J}(n)$的行秩必须是 n；也就是说，式(3.20)中 $\mathbf{J}(n)$的 n 行必须是线性无关的。不过，我们并不能保证这个条件总是满足。为了防止 $\mathbf{J}(n)$的秩亏损，通常的办法是给矩阵 $\mathbf{J}^T(n)\mathbf{J}(n)$加一个对角矩阵 $\delta\mathbf{I}$。参数 δ 是一个小的正常数，它的选择必须保证

$$\mathbf{J}^T(n)\mathbf{J}(n)+\delta\mathbf{I}:\text{对所有 } n \text{ 都是正定的}$$

在这个基础上，Gauss – Newton 方法以下面微小修正形式实现：

$$\mathbf{w}(n+1)=\mathbf{w}(n)-(\mathbf{J}^T(n)\mathbf{J}(n)+\delta\mathbf{I})^{-1}\mathbf{J}^T(n)\mathbf{e}(n) \tag{3.23}$$

当迭代次数 n 不断增大时，这个修正的影响是逐渐减少的。同时注意递归式(3.23)是修正的代价函数

$$\mathscr{E}(\mathbf{w})=\frac{1}{2}\left\{\delta\|\mathbf{w}-\mathbf{w}(n)\|^2+\sum_{i=1}^{n}e^2(i)\right\} \tag{3.24}$$

的解，其中 $\mathbf{w}(n)$是权值向量 $\mathbf{w}(i)$的当前值。

现在我们已经具备了解决线性自适应滤波器涉及的特殊问题所需的最优化工具。

3.4 线性最小二乘滤波器

同标题暗示的一样，一个线性最小二乘滤波器有两个明显的特征。第一，在它构造周围的神经单元是线性的，如图 3-1b 的模型所示。第二，用来设计滤波器的代价函数$\mathscr{E}(\mathbf{w})$是误差平方和，如式(3.17)的定义。在这个基础上，利用式(3.3)和(3.4)，误差向量 $\mathbf{e}(n)$可以表示如下：

$$\mathbf{e}(n)=\mathbf{d}(n)-[\mathbf{x}(1),\mathbf{x}(2),\cdots,\mathbf{x}(n)]^T\mathbf{w}(n)=\mathbf{d}(n)-\mathbf{X}(n)\mathbf{w}(n) \tag{3.25}$$

其中 $\mathbf{d}(n)$是 $n\times 1$ 的期望响应向量：

$$\mathbf{d}(n)=[d(1),d(2),\cdots,d(n)]^T$$

126

$\mathbf{X}(n)$是 $n\times m$ 的数据矩阵：

$$\mathbf{X}(n)=[\mathbf{x}(n),\mathbf{x}(2),\cdots,\mathbf{x}(n)]^T$$

由式(3.25)对 $\mathbf{w}(n)$取微分得到梯度矩阵

$$\nabla\mathbf{e}(n)=-\mathbf{X}^T(n)$$

相应地，$\mathbf{e}(n)$的 Jacobi 矩阵是

$$\mathbf{J}(n)=-\mathbf{X}(n) \tag{3.26}$$

因为误差式(3.19)对权值向量 $\mathbf{w}(n)$ 已是线性的，如下所示 Gauss-Newton 方法在一次迭代后收敛。将式(3.25)和(3.26)代入(3.22)得到

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)(\mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n)) \\ &= (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)\mathbf{d}(n)\end{aligned} \tag{3.27}$$

项 $(\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)$ 被看作是数据矩阵 $\mathbf{X}(n)$ 的*伪逆*(Golub and Van Loan(1996)，Haykin(1996))；即

$$\mathbf{X}^+(n) = (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n) \tag{3.28}$$

因此，我们可以把式(3.27)改写为紧凑的形式：

$$\mathbf{w}(n+1) = \mathbf{X}^+(n)\mathbf{d}(n) \tag{3.29}$$

这个公式表示下面陈述的一个简便方式："权值向量 $\mathbf{w}(n+1)$ 求解定义在持续时间为 n 的一个观察区间上的线性最小二乘问题。"

Wiener 滤波器：各态历经环境下的线性最小二乘滤波器的极限形式

一个有趣的情形是：当输入向量 $\mathbf{x}(i)$ 和期望响应 $d(i)$ 来自于各态历经(ergodic)平稳环境。我们可以用长期样本均值或时间均值来代替期望或总体均值(Gray and Davisson, 1986)。这样一个环境可以部分用以下二阶统计量来描述：

- 输入向量 $\mathbf{x}(i)$ 的相关矩阵(correlation matrix)；记为 $\mathbf{R}_x$
- 输入向量 $\mathbf{x}(i)$ 和期望响应 $d(i)$ 之间的互相关向量(cross-correlation vector)；记为 $\mathbf{r}_{xd}$。

这两个量分别定义如下：

127

$$\mathbf{R}_x = E[\mathbf{x}(i)\mathbf{x}^T(i)] = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}(n)\mathbf{x}^T(i) = \lim_{n\to\infty}\frac{1}{n}\mathbf{X}^T(n)\mathbf{X}(n) \tag{3.30}$$

$$\mathbf{r}_{xd} = E[\mathbf{x}(i)d(i)] = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}(i)d(i) = \lim_{n\to\infty}\frac{1}{n}\mathbf{X}^T(n)\mathbf{d}(n) \tag{3.31}$$

其中 E 表示统计期望算子。相应地，我们可以把式(3.27)的线性最小二乘解改写为：

$$\begin{aligned}\mathbf{w}_o &= \lim_{n\to\infty}\mathbf{w}(n+1) = \lim_{n\to\infty}(\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)\mathbf{d}(n) \\ &= \lim_{n\to\infty}\frac{1}{n}(\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\lim_{n\to\infty}\frac{1}{n}\mathbf{X}^T(n)\mathbf{d}(n) = \mathbf{R}_x^{-1}\mathbf{r}_{xd}\end{aligned} \tag{3.32}$$

这里 $\mathbf{R}_x^{-1}$ 是相关矩阵 $\mathbf{R}_x$ 的逆。为了纪念 Norbert Wiener 对这个问题作出的贡献，权值向量 $\mathbf{w}_o$ 称为线性最优滤波问题的 Wiener 解(Widrow and Stearns, 1985; Haykin, 1996)。因此，我们可以作出以下的陈述：

对一个各态历经过程，当观察样本数趋于无穷时，线性最小二乘滤波器渐进趋于 Wiener 滤波器。

设计 Wiener 滤波器需要二阶统计量的知识：输入向量 $\mathbf{x}(n)$ 的相关矩阵 $\mathbf{R}_x$ 和 $\mathbf{x}(n)$ 与期望响应 $d(n)$ 的互相关向量 $\mathbf{r}_{xd}$。但是，在实际遇到的很多重要情况下这些信息都是未知的。我们可以利用线性自适应滤波器(linear adaptive filter)来处理未知的环境，自适应在这里的意思是滤波器能够调整自己的自由参数来响应环境的统计变化。在连续基础上作这类调整的一个流行的算法是最小均方算法，它是与 Wiener 滤波器密切相关的。

3.5 最小均方算法

最小均方(least mean square, LMS)算法建立的基础是利用代价函数的瞬时值，即

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}e^2(n) \tag{3.33}$$

这里 $e(n)$ 是 n 时刻的测得的误差。把 $\mathscr{E}(\mathbf{w})$ 对权值向量 $\mathbf{w}$ 求导数得到

$$\frac{\partial \mathscr{E}(\mathbf{w})}{\partial \mathbf{w}} = e(n)\frac{\partial e(n)}{\partial \mathbf{w}} \tag{3.34}$$

128

如同在线性最小二乘滤波器上一样，LMS 算法运行在一个线性神经元，可以把误差信号表示为

$$e(n) = d(n) - \mathbf{x}^T(n)\mathbf{w}(n) \tag{3.35}$$

因此

$$\frac{\partial e(n)}{\partial \mathbf{w}(n)} = -\mathbf{x}(n)$$

和

$$\frac{\partial \mathscr{E}(\mathbf{w})}{\partial \mathbf{w}(n)} = -\mathbf{x}(n)e(n)$$

把后者作为梯度向量的一种估计，可以记

$$\hat{\mathbf{g}}(n) = -\mathbf{x}(n)e(n) \tag{3.36}$$

最后，利用式(3.36)作为式(3.12)中的最速下降法的梯度向量，可以写出 LMS 算法公式

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)e(n) \tag{3.37}$$

这里 η 是学习率参数。LMS 算法中围绕权值向量 $\hat{\mathbf{w}}(n)$ 的反馈环的作用就像一个低通滤波器，即通过误差信号的低频分量，而削弱高频分量(Haykin, 1996)。过滤动作的平均时间常数与学习率参数 η 成反比。因此，给 η 赋一个较小的值，自适应过程将进展缓慢。由此更多的过去数据被 LMS 算法记忆，导致一个更精确的过滤过程。换句话说，学习率参数 η 的倒数是 LMS 算法记忆的一种度量。

在式(3.37)中我们用 $\hat{\mathbf{w}}(n)$ 代替 $\mathbf{w}(n)$ 用来强调这样一个事实：利用最速下降法可以得到一个权值向量而 LMS 算法产生该权值向量的一个估计值。所以，使用 LMS 算法时我们牺牲掉最速下降法的一个明显特征。在最速下降法中，对一个给定的 η 权值向量 $\mathbf{w}(n)$ 在权值空间中有一个明确定义轨迹。对比之下，在 LMS 算法中权值向量 $\hat{\mathbf{w}}(n)$ 则跟踪一个随机的轨迹。由于这个原因，LMS 算法有时也被称为“随机梯度算法”。当 LMS 算法的迭代次数趋于无限时，$\hat{\mathbf{w}}(n)$ 在 Wiener 解 $\mathbf{w}_o$ 周围随机移动(布朗运动)。重要的事实是，不像最速下降法，LMS 算法不需要知道环境的统计特性。

129

在表 3-1 中小结 LMS 算法，它清楚表明这种算法的简单性。如这个表中表明的，对于算法的初始化，一般设算法中的权值向量初始值设为零。

表 3-1 LMS 算法小结

训练样本：	输入信号向量 $=\mathbf{x}(n)$ 期望响应 $=d(n)$
用户选择参数：	η
初始化：	设置 $\hat{\mathbf{w}}(n)=\mathbf{0}$
计算：	当 $n=1, 2, \cdots$，计算
$e(n)=d(n)-\hat{\mathbf{w}}^T(n)\mathbf{x}(n)$ $\hat{\mathbf{w}}(n+1)=\hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)e(n)$	

LMS 算法的信号流图表示

结合式(3.35)和(3.37)，我们可以把 LMS 算法中的权值向量演变过程表示如下：

$$\begin{aligned}\hat{\mathbf{w}}(n+1) &= \hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)[d(n) - \mathbf{x}^T(n)\hat{\mathbf{w}}(n)] \\ &= [\mathbf{I} - \eta\mathbf{x}(n)\mathbf{x}^T(n)]\hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)d(n)\end{aligned} \tag{3.38}$$

这里 $\mathbf{I}$ 是单位矩阵。通过运用 LMS 算法，我们认识到

$$\hat{\mathbf{w}}(n) = z^{-1}[\hat{\mathbf{w}}(n+1)] \tag{3.39}$$

这里 z^{-1} 是单位延迟操作符，意味着存储。利用式(3.38)和(3.39)，可以用图 3-3 描绘的信号流图表示 LMS 算法。这个信号流图揭示 LMS 算法是随机反馈系统的一个实例。反馈的出现对 LMS 算法的收敛有重要影响。

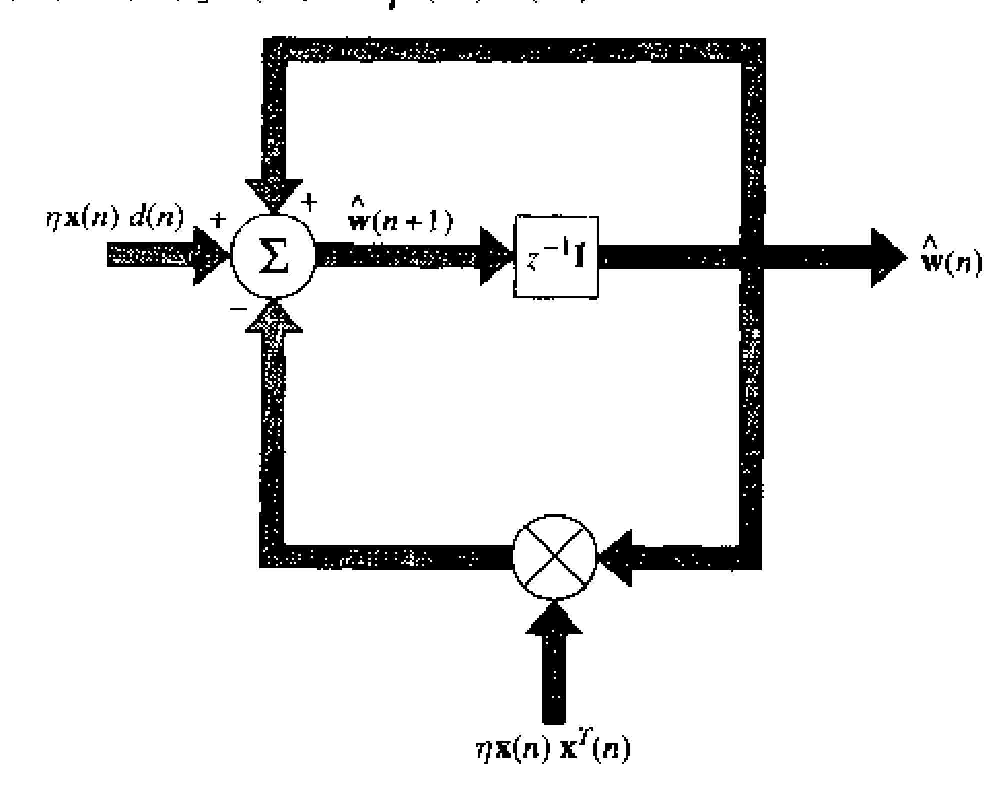


图 3-3 LMS 算法的信号流图表示

LMS 算法的收敛考虑

从控制论我们知道反馈系统的稳定性是由组成反馈环的参数决定的。[130] 从图 3-3 看出，正是较低的反馈环在 LMS 算法的运行中加入可变性。特别是，有两个不同的量，即学习率参数 η 和输入向量 $\mathbf{x}(n)$，决定这个反馈环的传输系数。因此我们推出输入向量 $\mathbf{x}(n)$的统计特征和学习率参数的取值影响 LMS 算法的收敛行为(即稳定性)。用不同的方法使用这个观察，可以陈述对于提供输入向量 $\mathbf{x}(n)$的特定环境，我们必须仔细选择学习率参数 η 以便使 LMS 算法收敛。

LMS 算法的第一个收敛准则是平均收敛，描述为

$$E[\hat{\mathbf{w}}(n)] \to \mathbf{w}_o \qquad \text{当 } n \to \infty \text{ 时} \tag{3.40}$$

这里 $\mathbf{w}_o$ 是 Wiener 解。不过，这样一个收敛准则没有多少实用价值，因为一系列零均值而其他为任意的随机向量在这种意义下是收敛的。

从实际情况考虑，真正的收敛应该是*均方收敛*，描述为

$$E[e^2(n)] \to \text{常数} \qquad \text{当 } n \to \infty \text{ 时} \tag{3.41}$$

但是，一个 LMS 算法的均方收敛的详细分析是相当复杂的。为了使这个分析在数学上可行，通常作出下列假设：

1. 顺序的输入向量 $\mathbf{x}(1)$，$\mathbf{x}(2)$，… 互相统计独立。

2. 在第 n 步，输入向量 $\mathbf{x}(n)$对以前样本的期望响应 $d(1)$，$d(2)$，…，$d(n-1)$是统计独立的。

3. 在第 n 步，期望响应 $d(n)$与 $\mathbf{x}(n)$有关，但对以前的所有期望响应统计独立。

4. 输入向量 $\mathbf{x}(n)$和期望响应 $d(n)$抽取自 Gauss 分布总体。

在此基础上的 LMS 算法统计分析称为*独立理论*(independence theory)(Widrow et al.,1976)。

通过引入独立理论原理并假设学习率参数 η 足够小，Haykin(1996)证明只要 η 满足条件

$$0 < \eta < \frac{2}{\lambda_{max}} \tag{3.42}$$

LMS 是均方收敛的，这里 λ_{max} 是相关矩阵 $\mathbf{R}_x$ 的最大特征值。但是，在 LMS 算法的典型应用中，λ_{max} 是未知的。为了克服这个困难，$\mathbf{R}_x$ 的迹(trace)被当作 λ_{max} 的保守估计，在这种情况下，等式(3.42)可以改写为

$$0 < \eta < \frac{2}{\mathrm{tr}[\mathbf{R}_x]} \tag{3.43}$$

131

这里 $\mathrm{tr}[\mathbf{R}_x]$ 表示矩阵 $\mathbf{R}_x$ 的迹。根据定义，一个方阵的迹等于其对角元素的和。因为相关矩阵 $\mathbf{R}_x$ 的对角元素等于相关传感器输入的均方值，我们可以重新表述 LMS 算法均方收敛的条件如下：

$$0 < \eta < \frac{2}{\text{传感器输入的均方值之和}} \tag{3.44}$$

如果学习率参数满足此条件，那么 LMS 算法也能保证平均收敛。就是说，均方收敛能推出平均收敛，但反过来不一定成立。

LMS 算法的优点和局限

正如表 3-1 算法概述中说明的那样，LMS 算法重要优点就是简单。此外，LMS 算法是模型独立的，因此是鲁棒的，这意味这小的模型不确定性和小的扰动(即小的能量扰动)只可能导致小的估计误差(误差信号)。用精确的数学术语，LMS 算法按照 H^∞ 准则(或最小最大准则)是最优的(Hassibi et al.,1993,1996)。在 H^∞ 意义下的最优性基本原理要对付最坏情况[4]：

如果你不知道你面对的是什么，计划最坏的情况并优化它。

长期以来 LMS 算法被当作梯度下降法的瞬时逼近。但是，LMS 的 H^∞ 最优性为这个广泛应用的算法提出了一个严格的基础。特别，它解释算法在稳定和不稳定环境下的令人满意工作的能力。这里"不稳定"环境是指统计特性随时间变化的环境。在这样一个环境下，最优的 Wiener 解随时间变化，LMS 算法现在有了一个附加任务——跟踪 Wiener 滤波器参数的变化。

LMS 算法的主要局限性是收敛速度较慢，并且对输入特征结构的变化反应较灵敏(Haykin，1996)。LMS 算法一般需要输入空间维数十倍的迭代次数才能达到稳定状态。当输入空间维数较高时缓慢的收敛速度会变得特别严重。至于对环境条件的变化反应很灵敏，LMS 算法对输入向量 $\mathbf{x}$ 的相关矩阵 $\mathbf{R}_x$ 的条件数或特征值散布的变化反应特别灵敏。$\mathbf{R}_x$ 的条件数记为 $\chi(\mathbf{R}_x)$，定义如下：

$$\chi(\mathbf{R}_x) = \frac{\lambda_{max}}{\lambda_{min}} \tag{3.45}$$

这里 λ_{max} 和 λ_{min} 分别是矩阵 $\mathbf{R}_x$ 的最大和最小特征值。当输入向量 $\mathbf{x}(n)$ 所属的训练样本是病态情况时，也就是当条件数 $\chi(\mathbf{R}_x)$[5] 较大，LMS 算法对条件数 $\chi(\mathbf{R}_x)$ 变化的灵敏变得特别严重。注意 LMS 算法的 Hessian 矩阵定义为代价函数 $\mathscr{E}(\mathbf{w})$ 对 $\mathbf{w}$ 的二阶导数，它等于相关矩阵 $\mathbf{R}_x$；请见习题 3.8。因此，在这里的讨论中，我们用 Hessian 矩阵替换相关矩阵 $\mathbf{R}_x$ 所有讨论仍然成立。

132

3.6 学习曲线

一个检验 LMS 算法或一个普通自适应滤波器的收敛行为的非正式方法是绘制滤波器在变化环境条件下的学习曲线。学习曲线是绘制估计误差的均方值$\mathscr{E}_{av}(n)$关于迭代次数 n 的图像。

设想一个试验涉及一个滤波器总体，每个滤波器在特殊算法控制下运行。假设算法的细节，包括初始化，对所有滤波器是一样的。滤波器之间的差异是来源于可用的训练样本的输入向量 $\mathbf{x}(n)$与期望响应 $d(n)$的抽取的随机方式不同。对每一个滤波器我们画出估计误差(即期望响应与实际滤波器输出之差)的平方值关于迭代次数的图像。一条样本学习曲线由噪声指数组成，噪声来源于滤波器固有的随机性。为了计算总体平均学习曲线(即$\mathscr{E}_{av}(n)$关于 n 的图像)，我们利用试验中滤波器总体的样本学习曲线的平均，从而平滑噪声的影响。

假设自适应滤波器是稳定的，我们发现总平均学习曲线是从由初始条件决定的一个很大的值$\mathscr{E}_{av}(0)$开始，然后以某种速率下降，此速率由滤波器的使用种类决定，最后收敛到一个稳定值$\mathscr{E}_{av}(\infty)$，如图 3-4 所示。在学习曲线的基础上我们能够定义自适应滤波器的收敛速率为$\mathscr{E}_{av}(n)$减少到任意一个选定值(例如原始值$\mathscr{E}_{av}(0)$的10%)所需的迭代次数 n。

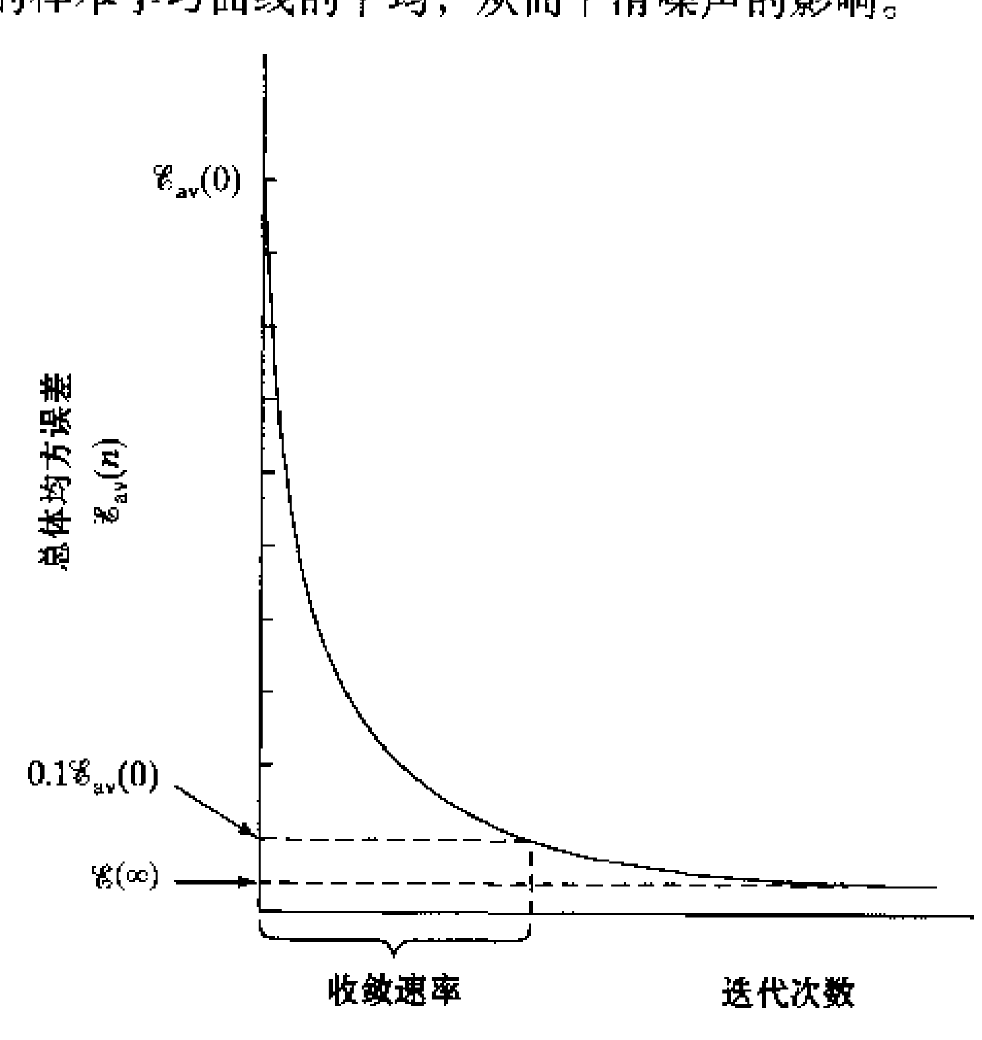


图 3-4 LMS 算法的理想学习曲线

另一个由学习曲线推出的有用的自适应滤波器特性是误调节(misadjustment)，记为$\mathcal{M}$。令$\mathscr{E}_{min}$表示 Wiener 滤波器产生的最小均方误差，它在已知相关矩阵 $\mathbf{R}_x$和互相关向量 $\mathbf{r}_{xd}$值的基础上设计。我们可以定义自适应滤波器的误调节如下(Widrow and Stearns, 1985; Haykin, 1996)：

$$\mathcal{M} = \frac{\mathscr{E}(\infty) - \mathscr{E}_{min}}{\mathscr{E}_{min}} = \frac{\mathscr{E}(\infty)}{\mathscr{E}_{min}} - 1 \tag{3.46}$$

误调节$\mathcal{M}$是一个无量纲的量，它用来衡量自适应滤波器在均方误差意义下和最优有多近。相对单位 1 来说$\mathcal{M}$越小，算法的自适应过滤行为就越精确。通常把误调节$\mathcal{M}$表示为百分比形式。所以，例如一个 10% 的误调节意味着自适应滤波器(在适应完成后)产生一个比相应的 Wiener 滤波器产生的最小均方误差$\mathscr{E}_{min}$大 10% 的均方误差。这种情况在实际中通常被认为是令人满意的。

133

另一个 LMS 算法的重要特性是稳定时间(settling time)。但是，对稳定时间并没有惟一的定义。例如，我们可以用具有给定平均时间常数 τ_{av}的单指数函数曲线来逼近学习曲线，然后利用所得的 τ_{av}当作稳定时间的粗略度量。τ_{av}值越小，稳定时间就越快(即 LMS 算法越快收

敛到"稳定"状态)。

对于一个较好程度的逼近来说，LMS 算法的误调节$\mathcal{M}$是与学习率参数 η 成正比的，而平均时间常数 τ_{av}是与学习率参数 η 成反比的(Widrow and Stearns,1985;Haykin,1996)。我们因此有这样一个矛盾的结果：如果降低学习率参数使得误调节下降，那么 LMS 算法的稳定时间将增加。反过来，如果增加学习率参数加速学习过程，那么误调节也增加。因此在设计 LMS 算法时对学习参数 η 的选择必须特别注意，这样才能得到一个满意的整体性能。

3.7 学习率退火进度

LMS 算法遇到的困难可归因于学习率参数在计算过程中保持不变，表示为

$$\eta(n) = \eta_0 \qquad \text{对所有 } n \tag{3.47}$$

这只是学习率参数假设最简单的可能形式。相反，在 Robbins 和 Monro 有关随机逼近的经典论文中(1951)，学习率参数是随时间改变的。在随机逼近文献中最常用到的学习率参数随时间变化的形式是 [134]

$$\eta(n) = \frac{c}{n} \tag{3.48}$$

这里 c 是常数。这样一个选择确实足够保证随机逼近算法的收敛性(Ljung,1977;Kushner and Clark,1978)。但是，当常数 c 较大时，对于较小的 n 参数有可能出现参数放大的危险。

作为等式(3.47)和(3.48)的替代物，我们可以使用 Darken and Moody(1992)定义的*搜寻后收敛进度*(search-then-converge schedule)

$$\eta(n) = \frac{\eta_0}{1 + (n/\tau)} \tag{3.49}$$

这里 η_0和 τ 是用户选择的常数。在自适应的早期阶段，即迭代次数 n 相对搜寻时间常数 τ 较小时，学习率参数 $\eta(n)$近似等于 η_0，算法运行实际上也是与"标准"LMS 算法一样的，如图 3-5 所示。因此，通过在允许范围内选择一个较大 η_0，我们希望对滤波器的可调权值能找到在一组较好的值并在其中上下浮动。然后，当迭代次数 n 比搜寻时间常数 τ 大时，学习率参数近似为 c/n，这里 $c=\tau\eta_0$，如图 3-5 所示。算法现在以一个传统的随机逼近算法运行，且权值收敛到它们的最优值。这样搜寻后收敛进度具有把标准 LMS 算法的期望特征和传统随机逼近理论结合起来的潜力。

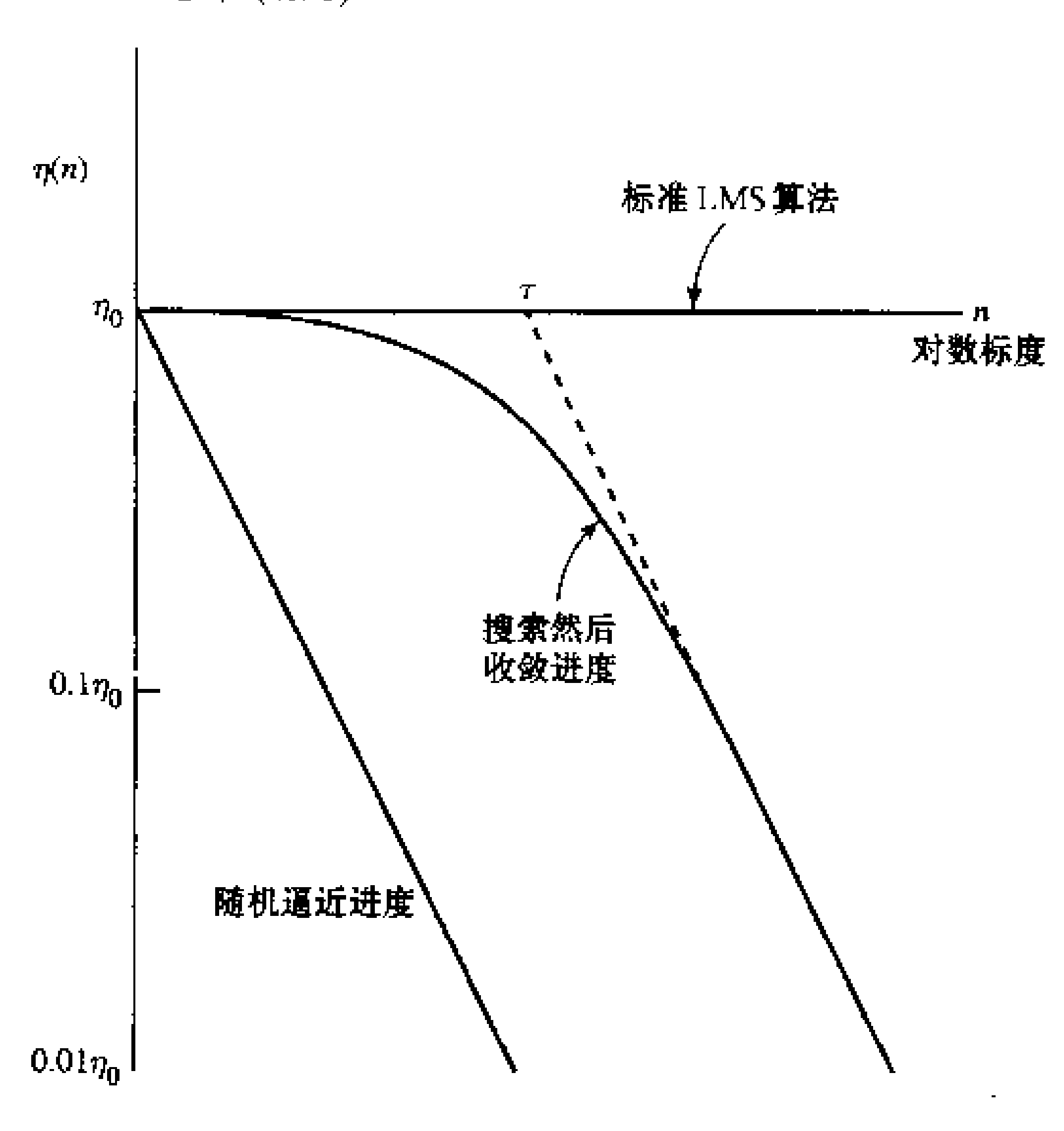


图 3-5 学习率退火进度

3.8 感知器

我们现在进入本章的第二部分，处理 Rosenblatt 的感知器，以后都简称为感知器(perceptron)。前面几节描述的 LMS 算法建立在一个线性神经元上，而感知器建立在一个非线性神经元上，即神经元的 McCulloch-Pitts 模型。我们回忆第1章里讲的这种神经元模型由一个线性组合器和随后的硬限幅器(执行一个符号函数)组成，如图3-6所示。神经元模型的求和节点计算应用于突触上的输入的一个线性组合，同时也合并一个外部的应用偏置。这个计算得出的和，也就是诱导局部域，被用到一个硬限幅器。于是当硬限幅器输入为正时，神经元输出 +1，反之则输出 -1。

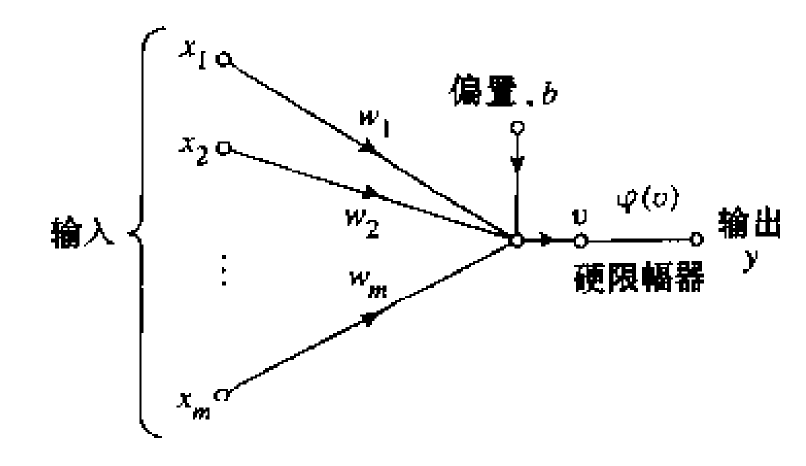


图3-6 感知器的符号流图

在图3-6的符号流图模型中，感知器的突触权值记为 w_1，w_2，…，w_m。相应地，用于感知器的输入量记为 x_1，x_2，…，x_m。外部应用偏置记为 b。从这个模型我们发现硬限幅器输入或神经元的诱导局部域是

135

$$v = \sum_{i=1}^{m} w_i x_i + b \tag{3.50}$$

感知器的目的是把外部应用刺激 x_1，x_2，…，x_m 正确分为$\mathscr{C}_1$ 和$\mathscr{C}_2$ 两类。分类规则是：如果感知器输出 y 是 +1 就将 x_1，x_2，…，x_m 表示的点分入类$\mathscr{C}_1$，如果感知器输出 y 是 -1 则分入$\mathscr{C}_2$。

为了进一步观察模式分类器的行为，一般要在 m 维信号空间中画出决策区域图，这个空间是由 m 个输入变量 x_1，x_2，…，x_m 张成的。在最简单的感知器中有被一个超平面分开的两个决策区域，此超平面定义为

136

$$\sum_{i=1}^{m} w_i x_i + b = 0 \tag{3.51}$$

对两个输入变量 x_1 和 x_2 的情况已在图3-7中举例说明，图中的决策边界是直线。位于边界线上方的点 (x_1, x_2) 分入$\mathscr{C}_1$ 类，位于边界线下方的点 (x_1, x_2) 分入$\mathscr{C}_2$ 类。注意这里偏置 b 作用仅仅把决策边界从原点移开。

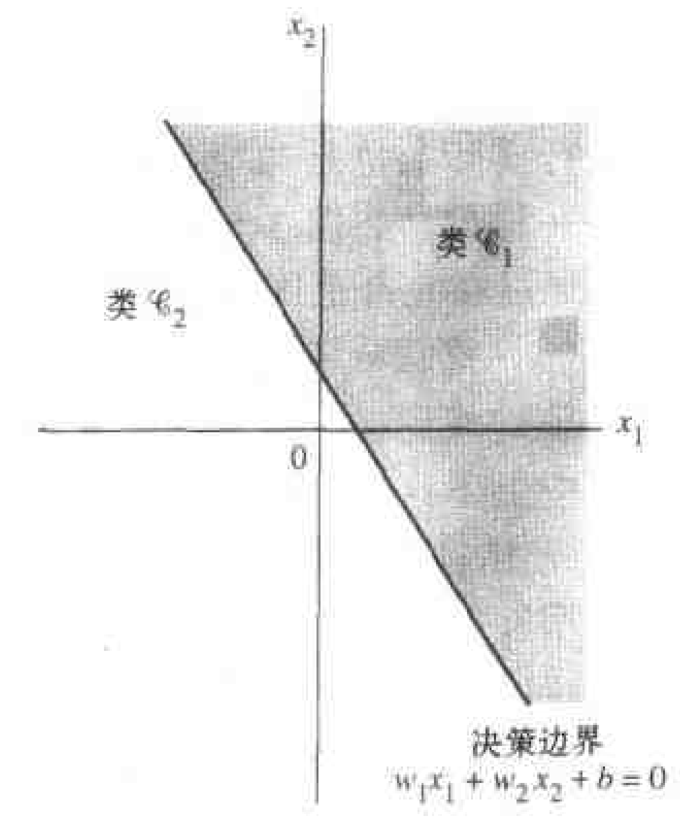


图3-7 作为一个两维两类模式分类问题的决策边界的一个超平面的实例(在这个例子中超平面是一条直线)

感知器的突触权值 w_1，w_2，…，w_m 可以通过多次迭代达到适应。对于自适应性我们可以使用通称为感知器收敛算法的误差修正规则。

3.9 感知器收敛定理

为了导出感知器误差修正学习算法，我们发现处理图3-8中的修改的信号流图更方便。在这个与图3-6中模型等价的第二个模型中，偏置 $b(n)$ 被当

作一个等于 +1 的固定输入量驱动的突触权值。我们因此定义 $(m+1)\times 1$ 输入向量

$$\mathbf{x}(n)=[+1,x_1(n),x_2(n),\cdots,x_m(n)]^T$$

这里 n 表示使用算法时的迭代步数。相应地我们定义 $(m+1)\times 1$ 权值向量

$$\mathbf{w}(n)=[b(n),w_1(n),w_2(n),\cdots,w_m(n)]^T$$

因此，线性组合器的输出可以写成紧凑形式

$$v(n)=\sum_{i=0}^{m}w_i(n)x_i(n)=\mathbf{w}^T(n)\mathbf{x}(n) \qquad (3.52)$$

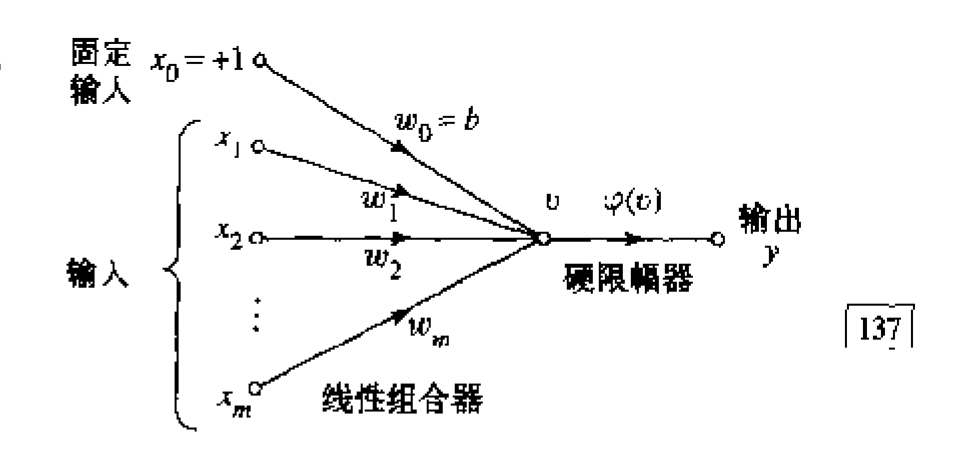


图 3-8 等价的感知器信号流图；为清楚起见省略了对时间的依赖性

137

这里 $w_0(n)$ 表示偏置 $b(n)$。对固定的 n，等式 $\mathbf{w}^T\mathbf{x}=0$ 在有关 x_1，x_2，…，x_m 的 m 维空间中(对某些规定偏置)定义了一个超平面，它就是两个不同输入类之间的决策平面。

为了感知器正确工作，$\mathscr{C}_1$ 和 $\mathscr{C}_2$ 两个类必须是线性可分的。这意味着待分类模式必须分离得足够开以保证决策平面是超平面。这个要求对两维感知器的情形如图 3-9 所示。在图 3-9a 中两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 分离得足够开，使得我们能画一个超平面(在此例中是一条直线)作为决策边界。但是，假如允许两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 靠得太近，如图 3-9b 所示，它们就变成非线性可分的，这种情况超出了感知器的计算能力。

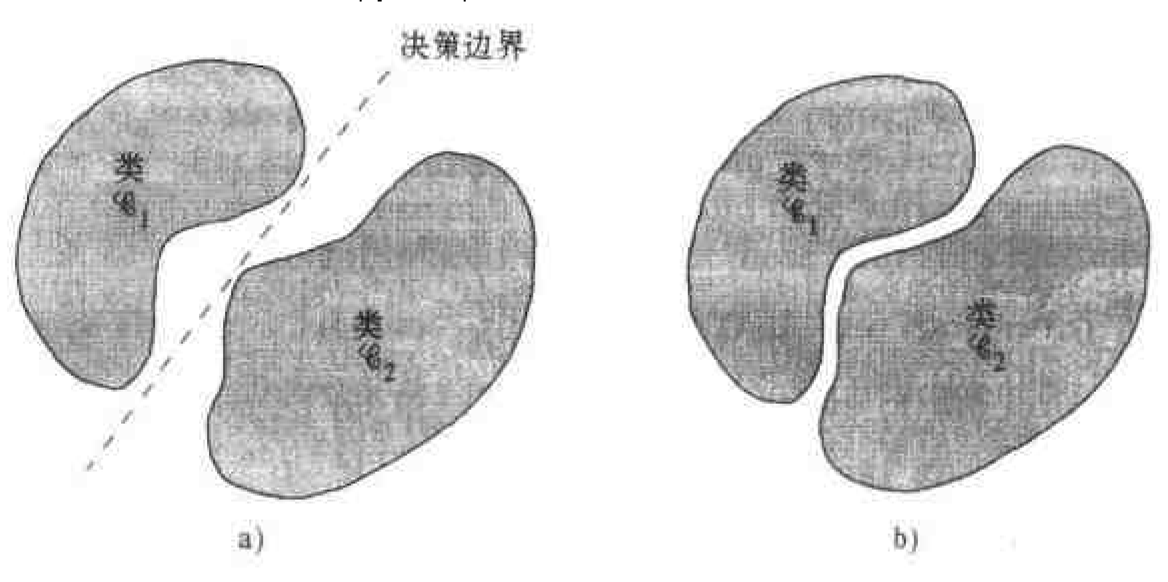


图 3-9
a)一对线性分离模式 b)一对非线性分离模式

假设感知器的输入变量来源于两个线性可分类。设 $\mathscr{X}_1$ 为训练向量 $\mathbf{x}_1(1)$，$\mathbf{x}_1(2)$，…中属于类 $\mathscr{C}_1$ 的向量组成的子集，$\mathscr{X}_2$ 表示训练向量 $\mathbf{x}_2(1)$，$\mathbf{x}_2(2)$，…属于类 $\mathscr{C}_2$ 的向量组成的子集。$\mathscr{X}_1$ 和 $\mathscr{X}_2$ 的并集是整个训练集 $\mathscr{X}$。给定向量集 $\mathscr{X}_1$ 和 $\mathscr{X}_2$ 来训练分类器，训练过程涉及对权值向量 $\mathbf{w}$ 的调整使得两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 线性可分。也就是，存在一个权值向量 $\mathbf{w}$ 具有以下性质

138

$$\begin{aligned}&\mathbf{w}^T\mathbf{x}>0 \quad \text{对属于类}\mathscr{C}_1\text{ 的每个输入向量 }\mathbf{x}\\&\mathbf{w}^T\mathbf{x}\leqslant 0 \quad \text{对属于类}\mathscr{C}_2\text{ 的每个输入向量 }\mathbf{x}\end{aligned} \qquad (3.53)$$

在式(3.53)的第二行中当 $\mathbf{w}^T\mathbf{x}=0$ 时我们随意地选择输入向量 $\mathbf{x}$ 属于类 $\mathscr{C}_2$。给定训练向量子集 $\mathscr{X}_1$ 和 $\mathscr{X}_2$，简单感知器的训练问题就是找到一个权值向量 $\mathbf{w}$ 满足式(3.53)中的两个不等式。

使基本感知器的权值向量自适应的算法现在可以用以下公式表述：

1. 假如训练集合的第 n 个成员 $\mathbf{x}(n)$ 根据算法中的第 n 次迭代的权值向量 $\mathbf{w}(n)$ 正确分类，那么感知器的权值向量按下述规则不会修改：

$$\begin{aligned} \mathbf{w}(n+1) &= \mathbf{w}(n) \quad \text{假如 } \mathbf{w}^T\mathbf{x}(n) > 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_1 \\ \mathbf{w}(n+1) &= \mathbf{w}(n) \quad \text{假如 } \mathbf{w}^T\mathbf{x}(n) \leqslant 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_2 \end{aligned} \tag{3.54}$$

2. 否则，感知器的权值向量根据以下规则更新：

$$\begin{aligned} \mathbf{w}(n+1) &= \mathbf{w}(n) - \eta(n)\mathbf{x}(n) \quad \text{假如 } \mathbf{w}^T(n)\mathbf{x}(n) > 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_2 \\ \mathbf{w}(n+1) &= \mathbf{w}(n) + \eta(n)\mathbf{x}(n) \quad \text{假如 } \mathbf{w}^T(n)\mathbf{x}(n) \leqslant 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_1 \end{aligned} \tag{3.55}$$

这里学习率参数 $\eta(n)$ 控制在第 n 次迭代中对权值向量的调节。

假如 $\eta(n)=\eta>0$，这里 η 是与迭代次数 n 无关的常数，我们有一个感知器的固定增量自适应规则。

后面我们首先证明当 $\eta=1$ 时固定增量自适应规则的收敛性。很明显 η 的具体值是不重要的，只要它是正的。$\eta\neq1$ 时的值不影响模式可分性而仅仅改变模式向量。对于 $\eta(n)$ 变化的情况稍后考虑。

给出的证明针对初始条件 $\mathbf{w}(0)=\mathbf{0}$。假设 $\mathbf{w}^T(n)\mathbf{x}(n)<0$ 对 $n=1, 2, \cdots$，且输入向量 $\mathbf{x}(n)$ 属于子集 $\mathscr{X}_1$。这样，既然式(3.53)的第二个的条件不满足，那么感知器不能正确地对向量 $\mathbf{x}(1)$，$\mathbf{x}(2)$，$\cdots$ 进行分类。在常量 $\eta(n)=1$ 的情况下，我们可以利用式(3.55)的第二行写作

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{x}(n) \quad \text{对 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_1 \tag{3.56}$$

给定初始条件 $\mathbf{w}(0)=\mathbf{0}$，我们可以迭代求解这个关于 $\mathbf{w}(n+1)$ 方程得到结果

139

$$\mathbf{w}(n+1) = \mathbf{x}(1) + \mathbf{x}(2) + \cdots + \mathbf{x}(n) \tag{3.57}$$

因为假设类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 为线性可分的，对属于子集 $\mathscr{X}_1$ 的向量 $\mathbf{x}(1)$，$\mathbf{x}(2)$，$\cdots$，$\mathbf{x}(n)$ 的不等式方程 $\mathbf{w}^T\mathbf{x}(n)>0$ 存在一个解 $\mathbf{w}_0$。对固定解 $\mathbf{w}_0$，我们可以定义一个正数 α，

$$\alpha = \min_{\mathbf{x}(n)\in\mathscr{X}_1} \mathbf{w}_0^T\mathbf{x}(n) \tag{3.58}$$

因此，在式(3.57)两边同乘行向量 $\mathbf{w}_0^T$，得到

$$\mathbf{w}_0^T\mathbf{w}(n+1) = \mathbf{w}_0^T\mathbf{x}(1) + \mathbf{w}_0^T\mathbf{x}(2) + \cdots + \mathbf{w}_0^T\mathbf{x}(n)$$

所以，依据等式(3.58)中的定义，我们有

$$\mathbf{w}_0^T\mathbf{w}(n+1) \geqslant n\alpha \tag{3.59}$$

下面利用众所周知的 Cauchy-Schwarz 不等式。给定两个向量 $\mathbf{w}_0$ 和 $\mathbf{w}(n+1)$，Cauchy-Schwarz 不等式表述为

$$\|\mathbf{w}_0\|^2 \|\mathbf{w}(n+1)\|^2 \geqslant [\mathbf{w}_0^T\mathbf{w}(n+1)]^2 \tag{3.60}$$

这里 $\|\cdot\|$ 表示所包含的变元向量的欧几里德范数，内积 $\mathbf{w}_0^T\mathbf{w}(n+1)$ 是标量。从式(3.59)得到 $[\mathbf{w}_0^T\mathbf{w}(n+1)]^2$ 大于或等于 $n^2\alpha^2$。从式(3.60)我们注意到 $\|\mathbf{w}_0\|^2\|\mathbf{w}(n+1)\|^2$ 大于或等于 $[\mathbf{w}_0^T\mathbf{w}(n+1)]^2$。这样得到

$$\|\mathbf{w}_0\|^2 \|\mathbf{w}(n+1)\|^2 \geqslant n^2\alpha^2$$

或等价地有

$$\|\mathbf{w}(n+1)\|^2 \geqslant \frac{n^2\alpha^2}{\|\mathbf{w}_0\|^2} \tag{3.61}$$

下面我们遵循另一种发展路线。特别地，可以把式(3.56)改写为

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{x}(k) \qquad 对\ k = 1,\cdots,n\ 且\ \mathbf{x}(k) \in \mathscr{X}_1 \tag{3.62}$$

通过对式(3.62)两边同取欧几里德范数的平方，我们得到

$$\| \mathbf{w}(k+1) \|^2 = \| \mathbf{w}(k) \|^2 + \| \mathbf{x}(k) \|^2 + 2\mathbf{w}^T(k)\mathbf{x}(k) \tag{3.63}$$

但是，在感知器对子集$\mathscr{X}_1$中的向量$\mathbf{x}(k)$不能进行正确分类的假设下，我们有$\mathbf{w}^T(k)\mathbf{x}(k) < 0$。因此从等式(3.63)中得到

$$\| \mathbf{w}(k+1) \|^2 \leqslant \| \mathbf{w}(k) \|^2 + \| \mathbf{x}(k) \|^2$$

或等价地有 $\| \mathbf{w}(k+1) \|^2 - \| \mathbf{w}(k) \|^2 \leqslant \| \mathbf{w}(k) \|^2, \qquad k = 1, \cdots, n$ (3.64) [140]

把$k = 1, \cdots, n$情况下的这些不等式和假设初始条件$\mathbf{w}(0) = \mathbf{0}$结合起来，得到不等式

$$\| \mathbf{w}(n+1) \|^2 \leqslant \sum_{k=1}^{n} \| \mathbf{x}(k) \|^2 \leqslant n\beta \tag{3.65}$$

这里β是一个正数，定义为

$$\beta = \max_{\mathbf{x}(k) \in \mathscr{X}_1} \| \mathbf{x}(k) \|^2 \tag{3.66}$$

式(3.65) 表明权值向量$\mathbf{w}(n+1)$的欧几里德范数平方的增长至多只能和迭代次数n形成线性关系。

当n有足够大的值时，式(3.65)的第二个结果显然是与式(3.61)的结果相矛盾。实际上，我们可以说n不能大于某个值n_{max}，值n_{max}使得式(3.61)和(3.65)等号都成立。这里，n_{max}是下面方程的解：

$$\frac{n_{max}^2 \alpha^2}{\| \mathbf{w}_0 \|^2} = n_{max}\beta$$

给定解向量$\mathbf{w}_0$，解出n_{max}，我们求出

$$n_{max} = \frac{\beta \| \mathbf{w}_0 \|^2}{\alpha^2} \tag{3.67}$$

这样我们证明了对所有的n，$\eta(n) = 1$，且$\mathbf{w}(0) = \mathbf{0}$，如果解向量$\mathbf{w}_0$存在，那么感知器权值的适应过程最多在$n_{max}$次迭代后终止。从式(3.58)，(3.66)和(3.67)注意到$\mathbf{w}_0$或n_{max}的解并不惟一。

我们现在可以叙述感知器的固定增量收敛定理(Rosenblatt, 1962)：

设训练向量的子集$\mathscr{X}_1$和$\mathscr{X}_2$是线性可分的，感知器的输入来自这两个子集。感知器在某个n_0次迭代后收敛，收敛是在如下意义下：

$$\mathbf{w}(n_0) = \mathbf{w}(n_0+1) = \mathbf{w}(n_0+2) = \cdots$$

是对$n_0 \leqslant n_{max}$的解向量。

下面考虑当$\eta(n)$变化时，单层感知器自适应的绝对误差修正过程。特别，设$\eta(n)$是满足下式的最小的整数：

$$\eta(n)\mathbf{x}^T(n)\mathbf{x}(n) > | \mathbf{w}^T(n)\mathbf{x}(n) |$$

利用这个过程我们发现如果第n次迭代时的内积$\mathbf{w}^T(n)\mathbf{x}(n)$存在符号错误，那么第$n+1$次迭代中$\mathbf{w}^T(n+1)\mathbf{x}(n)$符号就会是正确的。这说明如果$\mathbf{w}^T(n)\mathbf{x}(n)$有符号错误，我们可以通过设$\mathbf{x}(n+1) = \mathbf{x}(n)$来改变第$n+1$次迭代时的训练次序。换句话说，每个模式重复呈现给感知器直到模式正确分类。

[141] 注意当 $\mathbf{w}(0)$ 的初始值不为零时，仅仅是导致收敛需要的迭代次数或增加或减少，这依赖于 $\mathbf{w}(0)$ 与解 $\mathbf{w}_0$ 的相关程度。无论 $\mathbf{w}(0)$ 的值是多少，感知器都可以保证是收敛的。

在表 3-2 中我们对感知器收敛算法做出概述(Lippmann,1987)。在此表第三步计算感知器的实际响应中使用的记号 sgn(·)，表示符号函数(signum function)：

$$\operatorname{sgn}(v)=\begin{cases}+1 & 若\ v>0\\ -1 & 若\ v<0\end{cases} \tag{3.68}$$

这样我们可以把感知器的量化反应 $y(n)$ 表示为以下的简洁形式：

$$y(n)=\operatorname{sgn}(\mathbf{w}^T(n)\mathbf{x}(n)) \tag{3.69}$$

表 3-2 感知器收敛算法概述

变量和参数：

$$\begin{aligned}\mathbf{x}(n)&= m+1\ 维输入向量\\ &=[+1,x_1(n),x_2(n),\cdots,x_m(n)]^T\\ \mathbf{w}(n)&= m+1\ 维权值向量\\ &=[b(n),w_1(n),w_2(n),\cdots,w_m(n)]^T\\ b(n)&=偏置\\ y(n)&=实际响应(量化的)\\ d(n)&=期望响应\\ \eta&=学习率参数,一个比\ 1\ 小的正常数\end{aligned}$$

1. 初始化。设 $\mathbf{w}(0)=\mathbf{0}$。对时刻 $n=1,2,\cdots$ 执行下列计算。
2. 激活。在时间步 n，通过提供连续值输入向量 $\mathbf{x}(n)$ 和期望响应 $d(n)$ 来激活感知器。
3. 计算实际响应。计算感知器的实际响应：

$$y(n)=\operatorname{sgn}[\mathbf{w}^T(n)\mathbf{x}(n)]$$

这里 sgn(·)是符号函数。

4. 权值向量的自适应。更新感知器的权值向量：

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta[d(n)-y(n)]\mathbf{x}(n)$$

这里

$$d(n)=\begin{cases}+1 & 若\ \mathbf{x}(n)\ 属于类\ \mathscr{C}_1\\ -1 & 若\ \mathbf{x}(n)\ 属于类\ \mathscr{C}_2\end{cases}$$

[142] 5. 继续。时间步 n 增加 1，返回第 2 步。

注意输入向量 $\mathbf{x}(n)$ 是 $(m+1)\times 1$ 向量，它的第一个元素在整个计算中固定为 +1。相应地，权值向量 $\mathbf{w}(n)$ 是 $(m+1)\times 1$ 向量，它的第一个元素等于偏置 $b(n)$。表 3-2 中的另一个要点是：我们引入一个量化期望响应 $d(n)$，定义为

$$d(n)=\begin{cases}+1 & 若\ \mathbf{x}(n)\ 属于类\ \mathscr{C}_1\\ -1 & 若\ \mathbf{x}(n)\ 属于类\ \mathscr{C}_2\end{cases} \tag{3.70}$$

因此，权值向量 $\mathbf{w}(n)$ 的自适应是以误差修正学习规则(error-correction learning rule)形式下的累加：

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta[d(n)-y(n)]\mathbf{x}(n) \tag{3.71}$$

这里 η 是学习率参数，差 $d(n)-y(n)$ 扮演一个误差信号的角色。学习率参数是正常数，且 $0<\eta\leqslant 1$。当在这个区间里给 η 赋一个值时，我们必须记住两个互相冲突的需求(Lippmann, 1987)：

- 过去输入的平均值提供一个稳定的权值估计，这需要一个较小的 η

- 相对于产生输入向量 $\mathbf{x}$ 的过程的固有分布的实时变化，快速自适应需要较大的 η。

3.10 Gauss 环境下感知器与 Bayes 分类器的关系

感知器与一类通称 Bayes 分类器的经典模式分类器具有一定联系。在 Gauss 环境下，Bayes 分类器退化为一个线性分类器。这与感知器采用的形式是一样的。但是，感知器的线性特性并不是由于 Gauss 假设而具有的。这一节我们研究这种联系，并借此深入研究感知器的运行。我们首先对 Bayes 分类器作一个简单的复习。

Bayes 分类器

在 Bayes 分类器和 Bayes 假设检验过程中，我们最小化平均风险(记为$\mathcal{R}$)。对两类问题(记为类$\mathcal{C}_1$ 和类$\mathcal{C}_2$)，Van Trees(1968)定义的平均风险为：

$$\begin{aligned}\mathcal{R} = & c_{11}p_1\int_{\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\,d\mathbf{x} + c_{22}p_2\int_{\mathscr{X}_2} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2)\,d\mathbf{x} \\ & + c_{21}p_1\int_{\mathscr{X}_2} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\,d\mathbf{x} + c_{12}p_2\int_{\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2)\,d\mathbf{x}\end{aligned} \tag{3.72}$$

143

这里各项的定义如下：

p_i = 观察向量 $\mathbf{x}$(表示随机向量 $\mathbf{X}$ 的实现值)取自子空间$\mathscr{X}_i$ 的先验概率，这里 $i=1$，2 且 $p_1+p_2=1$。

c_{ij} = 当类$\mathcal{C}_j$ 是真实的类(即观察向量 $\mathbf{x}$ 是取自子空间$\mathscr{X}_j$)时决定支持由子空间$\mathscr{X}_i$ 代表的类$\mathcal{C}_i$ 的代价，$(i,j)=1$，2。

$f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_i)$ = 随机向量 $\mathbf{X}$ 的条件概率密度函数，假设观察向量 $\mathbf{x}$ 取自子空间$\mathscr{X}_i$，$i=1$，2。

式(3.72)右边的头两项表示正确决策(即正确分类)，从而最后两部分代表不正确决策(即错误分类)。每个决策通过两个因子乘积加权：作出决策的代价和发生的相对频率(即先验概率)。

目的在于确定一个最小化平均风险的策略。因为我们需要作出这样的决策，在全部观察空间$\mathscr{X}$中每个观察向量 $\mathbf{x}$ 必须被设定或者属于$\mathscr{X}_1$ 或者属于$\mathscr{X}_2$。因此

$$\mathscr{X} = \mathscr{X}_1 + \mathscr{X}_2 \tag{3.73}$$

相应地，我们可以把式(3.72)改写为等价的形式

$$\begin{aligned}\mathcal{R} = & c_{11}p_1\int_{\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\,d\mathbf{x} + c_{22}p_2\int_{\mathscr{X}-\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2)\,d\mathbf{x} \\ & + c_{21}p_1\int_{\mathscr{X}-\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\,d\mathbf{x} + c_{12}p_2\int_{\mathscr{X}_1} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2)\,d\mathbf{x}\end{aligned} \tag{3.74}$$

这里 $c_{11}<c_{21}$ 且 $c_{22}<c_{12}$。现在我们注意到下述事实：

$$\int_{\mathscr{X}} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\,d\mathbf{x} = \int_{\mathscr{X}} f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2)\,d\mathbf{x} = 1 \tag{3.75}$$

因此，式(3.74)变为

$$\mathcal{R} = c_{21}p_1 + c_{22}p_2 + \int_{\mathscr{X}_1}\left[p_2(c_{12}-c_{22})f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_2) - p_1(c_{21}-c_{11})f_{\mathbf{X}}(\mathbf{x}\mid\mathcal{C}_1)\right]d\mathbf{x} \tag{3.76}$$

式(3.76)右边的头两项代表一个固定代价。因为需要最小化平均风险$\mathcal{R}$，我们从式(3.76)得

到以下最优分类的策略：

1. 所有使被积函数（即方括号里的表达式）为负的观察向量 x 的值都归于子空间$\mathscr{X}_1$（即类 144 $\mathscr{C}_1$），因为此时积分对风险$\mathscr{R}$有一个负的贡献。

2. 所有使被积函数为正的观察向量 x 的值都必须从子空间$\mathscr{X}_1$中排除（即分配给类$\mathscr{C}_2$），因为此时积分对风险$\mathscr{R}$有一个正的贡献。

3. 使被积函数为零的 x 的值对平均风险$\mathscr{R}$没有影响，因此可以任意分配。我们假设这些点分配给子空间$\mathscr{X}_2$（即类$\mathscr{C}_2$）。

在这个基础上，我们写出 Bayes 分类器公式如下：

假如条件

$$p_1(c_{21} - c_{11})f_{\mathbf{X}}(\mathbf{x} \mid \mathscr{C}_1) > p_2(c_{12} - c_{22})f_{\mathbf{X}}(\mathbf{x} \mid \mathscr{C}_2)$$

满足，把观察向量 x 分配给子空间$\mathscr{X}_1$（即类$\mathscr{C}_1$）。否则把 x 分配给$\mathscr{X}_2$（即类$\mathscr{C}_2$）。

为了简化起见，定义

$$\Lambda(\mathbf{x}) = \frac{f_{\mathbf{X}}(\mathbf{x} \mid \mathscr{C}_1)}{f_{\mathbf{X}}(\mathbf{x} \mid \mathscr{C}_2)} \tag{3.77}$$

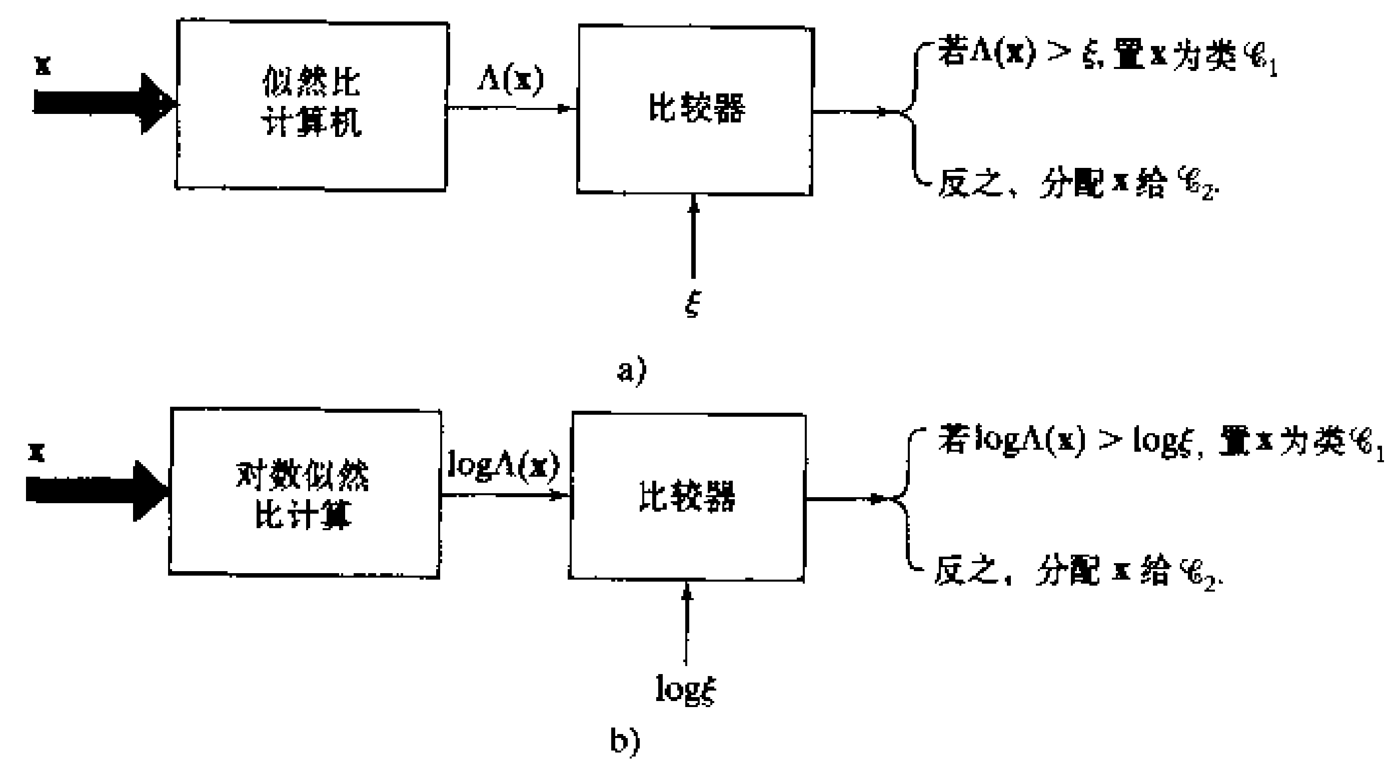
和

$$\xi = \frac{p_2(c_{12} - c_{22})}{p_1(c_{21} - c_{11})} \tag{3.78}$$

量 $\Lambda(\mathbf{x})$是两个条件概率密度函数的比，被称为似然比（likelihood ratio）。量 ξ 称为检验的阈值。注意 $\Lambda(\mathbf{x})$和 ξ 都是恒正的。根据这两个量，我们可以把 Bayes 分类重新表述为：

假如对一个观察向量 x，似然比 $\Lambda(\mathbf{x})$比阈值 ξ 大，就把 x 分配给类$\mathscr{C}_1$，反之，分配给类$\mathscr{C}_2$。

图 3-10a 是一个描绘 Bayes 分类器的模块图。此模块图的要点是两方面的：



145

图 3-10 Bayes 分类器的两个等价模型
a）似然比检验 b）对数似然比检验

1. 进行 Bayes 分类器设计的数据处理被完全限制在似然比 $\Lambda(\mathbf{x})$的计算中。

2. 此计算与分配给先验概率的值和决策过程中的代价是完全无关的。这两个量仅仅影响阈值 ξ。

从计算的观点，我们发现使用似然比的对数比使用似然比自身方便得多。这样做有两个理由。首先，对数是单调函数。其次，似然比 $\Lambda(\mathbf{x})$ 和阈值 ξ 都是正的。因此，Bayes 分类器可以以如图 3-10b 所示的等价形式实现。由于明显的原因，第二个图中嵌入的检验被称为对数似然比检验。

Gauss 分布下的 Bayes 分类器

现在考虑一个在 Gauss 分布下两类问题的特殊情形。随机向量 $\mathbf{X}$ 的均值依赖于 $\mathbf{X}$ 是属于类 $\mathscr{C}_1$ 还是 $\mathscr{C}_2$，但 $\mathbf{X}$ 的协方差阵对两类都是一样的。也就是说：

类 $\mathscr{C}_1$：$E[\mathbf{X}] = \boldsymbol{\mu}_1$

$$E[(\mathbf{X}-\boldsymbol{\mu}_1)(\mathbf{X}-\boldsymbol{\mu}_1)^T] = \mathbf{C}$$

类 $\mathscr{C}_2$：$E[\mathbf{X}] = \boldsymbol{\mu}_2$

$$E[(\mathbf{X}-\boldsymbol{\mu}_2)(\mathbf{X}-\boldsymbol{\mu}_2)^T] = \mathbf{C}$$

协方差矩阵 $\mathbf{C}$ 是非对角的，这意味着取自类 $\mathscr{C}_1$ 和类 $\mathscr{C}_2$ 的样本是相关的。假设 $\mathbf{C}$ 是非奇异的，这样它的逆矩阵 $\mathbf{C}^{-1}$ 存在。

在这个背景下我们可以把 $\mathbf{X}$ 的条件概率密度函数表示如下：

$$f_{\mathbf{X}}(\mathbf{x} \mid \mathscr{C}_i) = \frac{1}{(2\pi)^{m/2}(\det(\mathbf{C}))^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right), i = 1,2 \quad (3.79)$$

这里 m 是观察向量 $\mathbf{x}$ 的维数。

进一步假设

1. 两类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 的概率相同：

$$p_1 = p_2 = \frac{1}{2} \quad (3.80)$$

2. 错误分类造成同样的代价，正确分类的代价为零：

$$c_{21} = c_{12} \quad \text{和} \quad c_{11} = c_{22} = 0 \quad (3.81)$$

我们现在有了对两类问题设计 Bayes 分类器的信息。具体地，将式(3.79)代入(3.77)并取自然对数，我们得到(简化后)： 146

$$\begin{aligned}\log\Lambda(\mathbf{x}) &= -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_1)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_2)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_2) \\ &= (\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\mathbf{C}^{-1}\mathbf{x} + \frac{1}{2}(\boldsymbol{\mu}_2^T\mathbf{C}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^T\mathbf{C}^{-1}\boldsymbol{\mu}_1)\end{aligned} \quad (3.82)$$

把式(3.80)和式(3.81)代入式(3.78)并取自然对数，我们得到

$$\log\xi = 0 \quad (3.83)$$

式(3.82)和式(3.83)表明当前问题的 Bayes 分类器是线性分类器，如关系式

$$y = \mathbf{w}^T\mathbf{x} + b \quad (3.84)$$

所示，这里

$$y = \log\Lambda(\mathbf{x}) \quad (3.85)$$

$$\mathbf{w} = \mathbf{C}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \quad (3.86)$$

$$b = \frac{1}{2}(\boldsymbol{\mu}_2^T\mathbf{C}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^T\mathbf{C}^{-1}\boldsymbol{\mu}_1) \quad (3.87)$$

更进一步，分类器由一个权值向量 $\mathbf{w}$ 和偏置 b 的构成的线性组合器构成，如图3-11所示。

在式(3.84)的基础上，我们可以把对两类问题的对数似然比检验描述如下：

假如线性组合器(包括偏置 b)的输出是正的，把观察向量 $\mathbf{x}$ 分配给类 $\mathscr{C}_1$。否则，把它分配给类 $\mathscr{C}_2$。

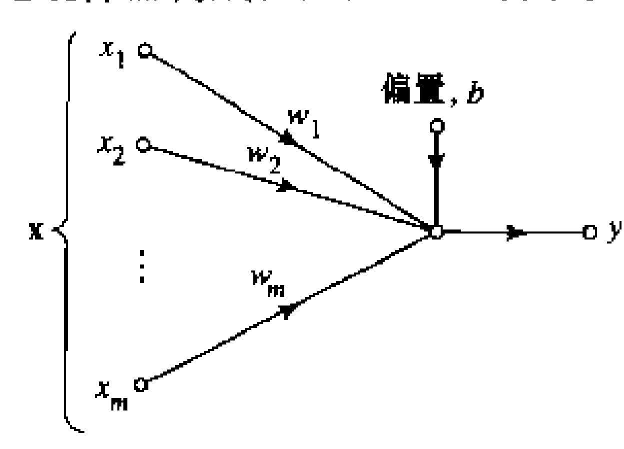


图3-11 Gauss分类器的信号流图

[147] 这里描述的Gauss环境下的Bayes分类器的运行与感知器是类似的，因为它们都是线性分类器；请见式(3.71)和(3.84)。但是，在它们之间还是有一些细微而且重要的不同，这必须被仔细检查(Lippmann, 1987)：

- 感知器运行的前提是待分模式是线性可分的。导出Bayes分类中假设两个Gauss分布的模式当然是互相重叠的，因此它们不是可分的。重叠的程度是由均值向量 $\boldsymbol{\mu}_1$ 和 $\boldsymbol{\mu}_2$ 以及协方差矩阵 $\mathbf{C}$ 决定的。重叠的性质如图3-12所示，这是对一个随机标量的特殊情况(即维数 $m=1$)。当输入如图所示是不可分且其分布是重叠的时候，感知器收敛算法出现一个问题，因为两类间的决策边界可能会持续振荡。
- Bayes分类最小化分类误差概率。这个最小化是与Gauss分布下两类之间的重叠无关。例如，在图3-12中的特例中，Bayes分类使决定边界总是位于Gauss分布下两类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 的交叉点上。
- 感知器收敛算法是非参数的，这指的是它没有关于固有分布形式的假设。它的运行是集中于发生在分布重叠地方的误差。当输入由非线性物理机制产生同时它们的分布是严重偏离而且非Gauss分布的时候，算法将工作得很好。相反，Bayes分类器是参数化的；它的导出是建立在Gauss分布的假设上的，这可能会限制它的适用范围。
- 感知器收敛算法是自适应的且实现简单；它的存储需求仅限于权值集合和偏置。另一方面，Bayes分类器设计是固定的；可以使它变成自适应的，但代价是增加存储量和更高计算复杂性。

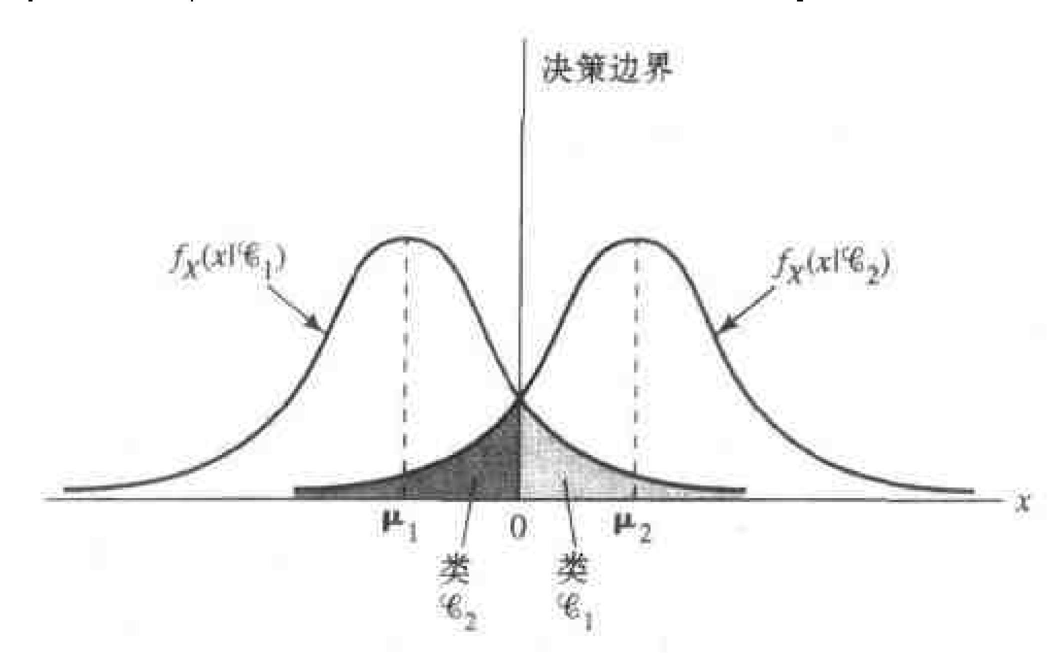


图3-12 两个重叠的一维Gauss分布

3.11 小结和讨论

感知器和使用LMS算法的自适应滤波器是本质上相关的，正如被它们的权值更新所表明的那样。实际上，它们代表基于误差修正学习的单层感知器的不同实现。术语“单层”用在这里是为了表示两者的计算层都是由单个神经元组成的——因此本章使用这个标题。[148] 但是，感知器和LMS算法在一些基本方面有区别：

- LMS 算法使用一个线性神经元，而感知器用神经元的 McCulloch-Pitts 形式模型。
- 感知器的学习过程执行有限步迭代后停止。相反，在 LMS 算法中发生持续学习，这是指当信号处理不停止时学习就不停止。

一个硬限幅器构成 McCulloch-Pitts 神经元的非线性元素。很容易提出这样一个问题：如果用一个 sigmoid 型非线性代替硬限幅器，感知器会不会有更好的表现？结果是不管我们使用硬限幅还是软限幅作为神经元模型中非线性源，感知器的状态稳定的决策特征基本不变(Shynk，1990；Shynk and Bershad，1991)。因此我们可以正式说只要限制为由线性组合器和随后一个非线性元素组成的神经元模型，不管非线性使用什么形式，一个单层感知器都只能在线性可分模式上进行模式分类。

我们用一个历史注释结束单层感知器的讨论。感知器和 LMS 算法大约在 20 世纪 50 年代晚期同时出现。LMS 算法真正经受住了时间的考验。实际上，它把自己作为一个自适应信号处理的主要工具(workhorse)，因为它实现起来较简便，应用效率也高。Rosenblatt 的感知器的重要性很大程度是在历史方面。

对于 Rosenblatt 感知器的第一个真正的批评是由 Minsky and Selfridge(1961)提出的。Minsky 和 Selfridge 指出，Rosenblatt 定义的感知器不管是用那种形式都不能推广到二进制数的奇偶校验对的情况，更不用说完成一般的抽象。Rosenblatt 感知器的计算局限后来又在 Minsky 和 Papert 的名著《感知器》中得到了严格的数学证明(1969，1988)。在给出一些出色的和非常详细的对感知器的数学分析以后，Minsky 和 Papert 证明，建立在局部学习例子基础上的 Rosenblatt 感知器从本质上无法进行全局的泛化。在他们著作的最后一章，Minsky 和 Papert 推测他们发现的 Rosenblatt 感知器的局限性对它的一种很特殊的变形——多层神经网络也是对的。下文摘录于他们著作(1969)的 13.2 节：

> 尽管(甚至由于!)它严重的局限，感知器展示了自身研究价值。它有很多吸引注意的优点：它的线性性，它迷人的学习法则，它清楚的作为一类并行计算范例的简单性。没有任何理由假定这些优点能带到多层感知器中。我们直觉判断推广到多层系统也不会有好结果，但是对于这一点我们认为证明(或否定)它是一个很重要的需要研究的问题。

这个结论在很大程度上导致了一个一直持续到 20 世纪 80 年代中期的对不仅是感知器而且是一般神经网络计算能力的严重怀疑。

但历史已经证明 Minsky 和 Papert 作出的推测似乎是不太公正的，因为我们现在已经有很多神经网络的高级形式，它们的计算能力比 Rosenblatt 感知器强得多。例如，第 4 章讨论的反向传播算法训练的多层感知器，第 5 章讨论的径向基函数网络，第 6 章讨论的支持向量机，都以它们各自的方法克服了单层感知器的计算局限性。 [149]

注释和参考文献

[1] Rosenblatt 预想的原始感知器模型的网络组织(1962)有三种类型的单元：感知单元，联想单元和响应单元。感知单元和联想单元之间的连接有固定的权值，而联想单元和响应单元之间的连接具有变化的权值。联想单元扮演的是设计成一个从环境输入中抽取模型的预处理器的角色。就仅关心可变权值而论，Rosenblatt 的原始感知器的运行与只有一个响应单元(即单个神经元)的特殊情况是基本一致的。

[2] **对一个向量的微分**

设 $f(\mathbf{w})$ 表示参数向量 $\mathbf{w}$ 的一个实值函数。$f(\mathbf{w})$ 对 $\mathbf{w}$ 的导数定义为如下向量：

$$\frac{\partial f}{\partial \mathbf{w}} = \left[\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \cdots, \frac{\partial f}{\partial w_m}\right]^T$$

这里 m 是向量 $\mathbf{w}$ 的维数。下面两种情形是很有用的：

情形 1 函数 $f(\mathbf{w})$ 定义为内积：

$$f(\mathbf{w}) = \mathbf{x}^T\mathbf{w} = \sum_{i=1}^{m} x_i w_i$$

因此，

$$\frac{\partial f}{\partial w_i} = x_i, \qquad i = 1,2,\cdots,m$$

或等价地，以矩阵形式表示：

$$\frac{\partial f}{\partial \mathbf{w}} = \mathbf{x} \tag{1}$$

情形 2 函数 $f(\mathbf{w})$ 定义为二次型：

$$f(\mathbf{w}) = \mathbf{w}^T\mathbf{R}\mathbf{w} = \sum_{i=1}^{m}\sum_{j=1}^{m} w_i r_{ij} w_j$$

这里 r_{ij} 是 $m \times m$ 矩阵 $\mathbf{R}$ 的第 ij 个元素。因此，

$$\frac{\partial f}{\partial w_i} = \sum_{j=1}^{m} r_{ij} w_j, \qquad i = 1,2,\cdots,m$$

或等价地，以矩阵形式表示：

150

$$\frac{\partial f}{\partial \mathbf{w}} = \mathbf{R}\mathbf{w} \tag{2}$$

式(1)和(2)为向量的实值函数的微分提供了两个有用的规则。

[3] **正定矩阵**

一个 $m \times m$ 矩阵 $\mathbf{R}$ 被称为是非负定的，如果它满足条件

$$\mathbf{a}^T\mathbf{R}\mathbf{a} \geq 0 \qquad 对任意\ \mathbf{a} \in \mathbb{R}^m$$

假如条件中的不等式满足，矩阵 $\mathbf{R}$ 被称为是正定的。

正定矩阵 $\mathbf{R}$ 的一个很重要的性质是它是非奇异的，因此逆矩阵 $\mathbf{R}^{-1}$ 存在。

正定矩阵 $\mathbf{R}$ 的另一个重要的性质是它的特征值或特征方程 $\det(\mathbf{R}) = 0$ 的根全部为正。

[4] **鲁棒性**

H^∞ 准则是由 Zames(1981)定义的，并在 Zames and Francis(1983)进一步发展。Doyle et al.(1989)，Green and Limebeer(1995)和 Hassibi et al.(1998)也对这个准则进行了讨论。

[5] 为了克服 LMS 算法的局限性，即收敛速度较慢和对相关矩阵 $\mathbf{R}_x$ 的条件数变化反应过于灵敏，我们可以使用递归最小二乘(recursive least-squares, RLS)算法，它利用我们在 3.4 节中描述的线性最小二乘滤波器进行递归实现。RLS 算法是 Kalman 滤波器的一个特例，后者被认为是非稳定环境下最优的线性滤波器。更重要的是，Kalman 滤波器计算利用所有过去扩展的数据并包含进行计算时的时间常数。关于 RLS 算法以及 RLS 算法和 Kalman 滤波器的关系的更多细节，参见 Haykin(1996)。Kalman 滤波器将在第 15 章中讨论。

习题

无约束最优化

3.1 研究包含一个权值 w 的最速下降法，考虑下列代价函数：

$$\mathscr{E}(w) = \frac{1}{2}\sigma^2 - r_{xd}w + \frac{1}{2}r_x w^2$$

这里 σ^2，r_{xd} 和 r_x 都是常数。

3.2 考虑代价函数

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}\sigma^2 - \mathbf{r}_{xd}^T\mathbf{w} + \frac{1}{2}\mathbf{w}^T\mathbf{R}_x\mathbf{w}$$

这里 σ^2 是常数，且

$$\mathbf{r}_{xd} = \begin{bmatrix} 0.8182 \\ 0.354 \end{bmatrix}, \mathbf{R}_x = \begin{bmatrix} 1 & 0.8182 \\ 0.8182 & 1 \end{bmatrix}$$

(a)求使$\mathscr{E}(\mathbf{w})$达到最小的最优值 $\mathbf{w}^*$。

(b)对下列两个学习率参数用最速下降法计算 $\mathbf{w}^*$：

(i)$\eta = 0.3$

(ii)$\eta = 1.0$

151

对每一种情况，画出权值向量 $\mathbf{w}(n)$在 W-平面演化产生的轨迹。

提示：(b)部分中情形(i)和情形(ii)的轨迹应与图 3-2 中的图形对应。

3.3 考虑式(3.24)的代价函数，它作为式(3.17)中定义的误差平方的和的修正形式，证明 Gauss-Newton 方法对式(3.24)中的应用是产生式(3.23)描述的权值更新。

LMS 算法

3.4 LMS 算法中输入向量 $\mathbf{x}(n)$的相关矩阵 $\mathbf{R}_x$ 定义为

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

定义 LMS 算法在均方收敛下的学习率参数 η 的取值范围。

3.5 正规化 LMS 算法通过以下对权值向量的递归形式表示：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\eta}{\|\mathbf{x}(n)\|^2}e(n)\mathbf{x}(n)$$

这里 η 是正常数且 $\|\mathbf{x}(n)\|$ 是输入向量 $\mathbf{x}(n)$的欧几里德范数。误差 $e(n)$定义为

$$e(n) = d(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n)$$

这里 $d(n)$是期望响应。为了使正规化 LMS 算法均方收敛，证明

$$0 < \eta < 2$$

3.6 LMS 算法用来实现广义旁瓣消除器，如图 2-16 所示。建立系统运行的方程，假设神经网络使用的是单个神经元。

3.7 考虑一个由样本 $x(n-1)$，$x(n-2)$，…，$x(n-m)$组成的输入向量的线性预测器，这里 m 是预测阶数。要求利用 LMS 算法得到输入样本 $x(n)$的预测 $\hat{x}(n)$。建立用来计算预测器的抽头权值 w_1，w_2，…，w_m 的递归关系式。

3.8 作为误差平方和副本的总体均值被看作代价函数，它是下面误差信号的均方值：

152

$$J(\mathbf{w}) = \frac{1}{2}E[e^2(n)] = \frac{1}{2}E[(d(n) - \mathbf{x}^T(n)\mathbf{w})^2]$$

(a)假设输入向量 $\mathbf{x}(n)$和期望响应 $d(n)$来自一个稳定环境，证明

$$J(\mathbf{w}) = \frac{1}{2}\sigma_d^2 - \mathbf{r}_{xd}^T\mathbf{w} + \frac{1}{2}\mathbf{w}^T\mathbf{R}_x\mathbf{w}$$

这里

$$\sigma_d^2 = E[d^2(n)]$$

$$\mathbf{r}_{xd} = E[\mathbf{x}(n)d(n)]$$

$$\mathbf{R}_x = E[\mathbf{x}(n)\mathbf{x}^T(n)]$$

(b)对这个代价函数，证明梯度向量和 $J(\mathbf{w})$的 Hessian 矩阵分别为如下形式：

$$\mathbf{g} = -\mathbf{r}_{xd} + \mathbf{R}_x\mathbf{w}$$

$$\mathbf{H} = \mathbf{R}_x$$

(c)在 LMS/Newton 算法中梯度向量 $\mathbf{g}$ 可以被它的瞬时值替代(Widrow and Stearns, 1985)。证明采用学习率参数 η 时这种算法可以表示如下：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\mathbf{R}_x^{-1}\mathbf{x}(n)(d(n) - \mathbf{x}^T(n)\mathbf{w}(n))$$

相关矩阵 $\mathbf{R}_x$ 的逆假设为正定的，是事先计算好的。

3.9 在此题中我们重新访问在2.11节中讨论的相关矩阵记忆。这个记忆的一个缺点是当为它提供一个关键模式 $\mathbf{x}_j$ 时，记忆体产生的实际响应 $\mathbf{y}$ 不能足够(在欧几里德的意义下)靠近期望响应(记忆模式)$\mathbf{y}_j$ 以便记忆可以很好联想。这个缺点是因为 Hebb 学习固有的，它没有利用从输出到输入的反馈。为了补救这个缺点，我们可以在记忆设计中耦合一个误差修正机制，迫使它恰当联想(Anderson, 1983)。

设 $\hat{\mathbf{M}}(n)$为误差修正学习过程第 n 次迭代学习的记忆矩阵。记忆矩阵 $\hat{\mathbf{M}}(n)$学习由联想表示的信息如下：

$$\mathbf{x}_k \rightarrow \mathbf{y}_k, \qquad k = 1,2,\cdots,q$$

(a)采用 LMS 算法解决这个问题，证明记忆矩阵的更新值定义为

$$\hat{\mathbf{M}}(n+1) = \hat{\mathbf{M}}(n) + \eta[\mathbf{y}_k - \hat{\mathbf{M}}(n)\mathbf{x}_k]\mathbf{x}_k^T$$

这里 η 是学习率参数。

(b)对自联想，$\mathbf{y}_k = \mathbf{x}_k$。对这个特例，证明当迭代次数 n 趋于无穷时，记忆自联想得很好，即证明

153

$$\mathbf{M}(\infty)\mathbf{x}_k = \mathbf{x}_k, \qquad k = 1,2,\cdots,q$$

(c)在(b)中的结果可以被看作一个特征值问题。在这个关系下，$\mathbf{x}_k$ 表示 $\mathbf{M}(\infty)$的一个特征向量。求 $\mathbf{M}(\infty)$的特征值。

3.10 此题中我们研究偏置对一个相关矩阵条件数的影响以及 LMS 算法的性能。

考虑一个随机向量 $\mathbf{X}$，它的协方差矩阵为

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

均值为 $\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$

(a)计算协方差矩阵 $\mathbf{C}$ 的条件数。

(b)计算相关矩阵 $\mathbf{R}$ 的条件数。

评价偏置 $\boldsymbol{\mu}$ 对 LMS 算法性能的影响。

Rosenblatt 的感知器

3.11 此题中，我们考虑另一种导出 Rosenblatt 感知器更新公式的方法。定义感知器准则函数(Duda and Hart, 1973)

$$J_p(\mathbf{w}) = \sum_{\mathbf{x} \in \mathscr{X}(\mathbf{w})} (-\mathbf{w}^T\mathbf{x})$$

这里$\mathscr{X}(\mathbf{w})$表示根据权值向量 $\mathbf{w}$ 的选择错误分类的样本集。注意，如果没有错误分类样本，$J_p(\mathbf{w})$定义为零，且假如 $\mathbf{w}^T\mathbf{x} \leq 0$ 输出是错误分类的。

(a)几何上证明 $J_p(\mathbf{w})$是与错误分类样本到决策边界的欧几里德距离的和成比例的。

(b)求 $J_p(\mathbf{w})$对权值向量 $\mathbf{w}$ 的梯度。

(c)利用(b)中得到的结果，证明感知器的权值更新是

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n) \sum_{\mathbf{x} \in \mathscr{X}(\mathbf{w}(n))} \mathbf{x}$$

这里$\mathscr{X}(\mathbf{w}(n))$表示用权值向量 $\mathbf{w}(n)$错误分类的样本集，且 $\eta(n)$是学习率参数。证明此结果对单样本修正的情形与式(3.54)和(3.55)描述的情形是基本一致的。

3.12 证明总结感知器收敛算法的式(3.68)至(3.71)是与式(3.54)和(3.55)一致的。

3.13 考虑两个一维 Gauss 分布类$\mathscr{C}_1$ 和$\mathscr{C}_2$，它们的方差均为 1。它们的均值为

$$\mu_1 = -10$$

$$\mu_2 = +10$$

这两个类本质上是线性可分的。设计一个分类器来分离这两个类。

3.14 假设图 3-6 中的信号流图的硬限幅器被如下 sigmoid 非线性替代：

$$\varphi(v) = \tanh\left(\frac{v}{2}\right)$$

这里 v 是诱导局部域。感知器的分类决策定义如下： [154]

如果输出 $y > \theta$ 观察向量 $\mathbf{x}$ 属于类$\mathscr{C}_1$，这里 θ 是阈值；反之，$\mathbf{x}$ 属于$\mathscr{C}_2$。

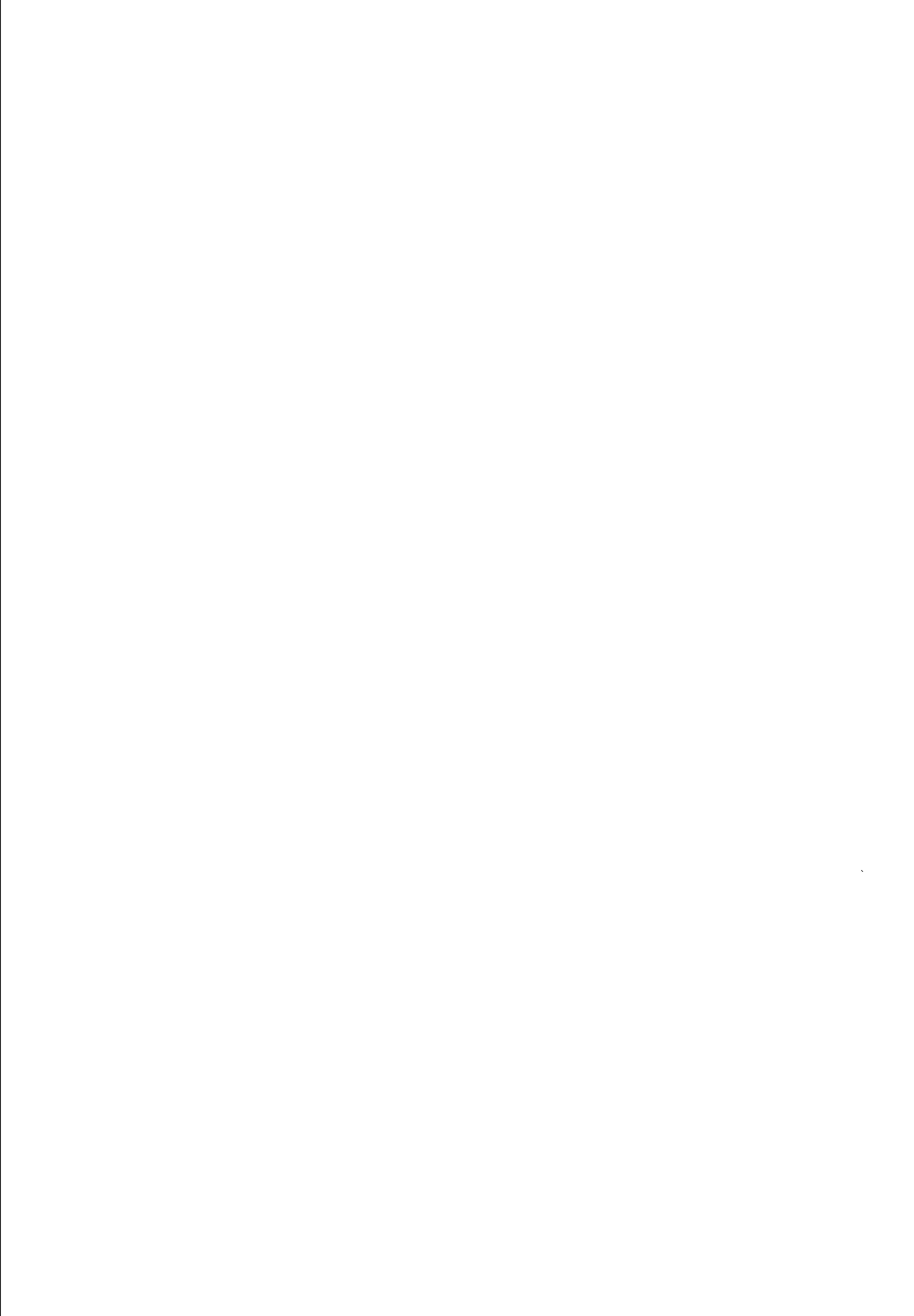
3.15 (a)感知器可以用来执行很多逻辑函数。证明它对二进制逻辑函数与(AND)、或(OR)和非(COMPLEMENT)的实现。

(b)感知器的一个基本局限是不能执行异或(XOR)函数。解释造成这个局限的原因。

3.16 式(3.86)和(3.87)定义 Bayes 分类在 Gauss 环境下的权值向量和偏置。当协方差矩阵 $\mathbf{C}$ 由

$$\mathbf{C} = \sigma^2\mathbf{I}$$ [155]

定义时，求此分类器的构成，这里 σ^2 是常数。



第4章　多层感知器

4.1　简介

在这一章我们学习多层前馈网络，它为神经网络的重要一类。这种网络典型地由三部分组成：一组感知单元(源节点)组成*输入层*，一层或多层计算节点的*隐藏层*，还有一层计算节点的*输出层*。输入信号在层层递进基础上前向传播通过网络。这些神经网络通常被称为*多层感知器*(multilayer perceptrons,MLPs)，它代表第3章考虑的单层感知器的推广。

在监督学习的方式下使用通称为*误差反向传播算法*这种非常普遍的算法训练多层感知器，它们已经成功应用于不同的复杂而困难的问题。误差反向传播算法是基于*误差修正学习*规则的。因此，它可以被看成是同样普遍使用的自适应滤波算法的推广：在第3章描述的用于单个神经元情形常用的最小均值平方(LMS)算法。

基本上，误差反向传播学习由两次经过网络不同层的通过组成：一次*前向通过*和一次*反向通过*。在前向通过中，一个活动模式(输入向量)作用于网络感知节点，它的影响经过网络一层接一层地传播。最后，产生一个输出作为网络的实际响应。在前向通过中，网络的突触权值全为固定的。另一方面，在*反向通过*中，突触权值全部根据误差修正规则来调整。特别是从目标响应减去网络的实际响应而产生误差信号。这个误差信号反向传播经过网络，与突触连接方向相反——因此叫“误差反向传播”。突触权值被调整使得网络的实际响应从统计意义上接近目标响应。误差反向传播算法在文献中称为*反向传播算法*(back-propagation algorithm)，或是简单称为*反向传播*(back-prop)。今后我们把它称为*反向传播算法*。由算法执行的学习过程被称之为*反向传播学习*。

156

多层感知器有三个突出的特点：

1. 网络中的每个神经元模型包括一个非线性激活函数。在这里要强调的非常重要一点是，与Rosenblatt感知器使用的硬限幅函数相反，非线性是光滑的(即处处可微)。满足非线性要求的一个普遍应用形式是由logistic函数

$$y_j = \frac{1}{1 + \exp(-v_j)}$$

定义的sigmoid非线性[1]，其中v_j是神经元j的诱导局部域(即所有突触输入的加权和减去偏置)，y_j是神经元j的输出。非线性的出现是很重要的，否则网络的输入输出关系会被归结为单层感知器所具有。而且，logistic函数的使用是基于生物学上考虑，因为它想说明真正神经元的反拗期(refra ctory)阶段。

2. 网络包括一层或多层隐藏神经元，它们不是网络输入输出的部分。这些隐藏层神经元逐步从输入模式(向量)中提取更多的有用特征，可以使网络学习复杂的任务。

3. 网络展示出高度的连接性，它由网络突触决定。网络连接的改变需要突触连接数量或其权值的改变。

正是由上述特性以及通过训练从经验中学习的能力相结合使得多层感知器具有它的计算

能力。然而，同样这些特性导致现阶段关于网络行为的知识的缺乏。首先，由于非线性分布式的存在和网络的高度连接性使得多层感知器的理论分析难于进行。第二，隐藏层的使用使得学习过程变得更不可想像。就间接的意义而言，学习过程必须决定输入模式的哪些特征应该由隐藏层神经元表示出来。学习过程因此变得更困难了，因为不得不在大得多的可能函数空间中搜索，同时不得不在输入模式的不同表示中进行选择(Hinton，1989)。

"反向传播"这个词的使用出现在1985年后，而它的广泛使用是在《*Parallel Distributed Processing*》(Rumelhart and McClelland，1986)这本书出版以后。关于反向传播算法的历史注释，请看1.9节。

反向传播算法的发展是神经网络发展史上的一个里程碑，因为它为训练多层感知器提供了一个有效的计算方法。虽然我们不能说反向传播算法为所有待解决的问题都提供了最优解，但是它使多层机器的学习前景不再和Minsky和Papert在其1969年所着的书中所暗示的那样悲观。

157

本章的组织

在本章中，我们学习多层感知器的基本知识以及反向传播学习。本章有七个部分。第一部分从4.2节到4.6节，讨论与反向传播学习有关的问题。在4.2节为引出反向传播算法作一些初步的铺垫。在4.3节用微分的链式规则详细导出该算法；在给出的推导中采用传统的方法。在4.4节对算法提出一个概述。在4.5节通过解决XOR问题这个例子说明如何使用反向传播算法，XOR问题是一个有趣的问题，但用单层感知器是无法解决的。在4.6节，为了反向传播算法实现得更好我们给出一些启发式方法或实际的指导方针。

第二部分从4.7节到4.9节，讨论多层感知器在模式识别中的用途。在4.7节介绍使用多层感知器解决统计模式识别问题的规则。在4.8节用一个计算机实验作为实例阐述反向传播学习应用于区分具有二维重叠Gauss分布的两类情况。在4.9节讨论隐藏层神经元作为特征检测器的重要作用。

本章的第三部分，包括从4.10节到4.11节，处理误差曲面的问题。在4.10节讨论反向传播学习在计算逼近函数偏导数中的重要作用。然后在4.11节讨论与误差曲面的Hessian矩阵相联系的计算问题。

第四部分，我们处理与用反向传播算法训练过的多层感知器性能有关的各种问题。在4.12节讨论泛化问题，它是关于学习的一个非常本质的问题。在4.13节讨论通过多层感知器得到的连续函数的逼近。在4.14节讨论把交叉确认作为统计设计的工具。在4.15节描述用一个程序有序地修剪一个多层感知器而同时使其整体性能至少保持不变(和不断提高)。当计算复杂性是首要关心的问题时，网络修剪就成为必要的。

第五部分完成反向传播学习的研究。4.16节总结反向传播学习的重要优点和局限。4.17节研究启发式方法，它为如何加速反向传播学习的收敛速率提供一个指导方针。

第六部分我们用一种不同的观点来看待学习。以提高学习为目的，在4.18节讨论监督学习作为一个数值优化问题的话题。特别地，我们描述用于监督学习的共扼梯度方法和拟Newton方法。

这一章最后一部分4.19节讨论多层感知器本身。在那里我们讨论一种有趣的神经网络结构——卷积多层感知器。这种网络已经成功用于解决困难的模式识别问题。

158

在4.20节以一些一般性讨论作为本章结束。

4.2 预备知识

图 4-1 表示一个具有两个隐藏层和一个输出层的多层感知器的结构图。为了构筑多层感知器一般形式的描述平台，这里说的网络是全连接的。这就是说在任意层上的一个神经元与它之前的层上的所有节点/神经元都连接起来。信号在一层接一层的基础上逐步流过，方向是向前的，从左到右。

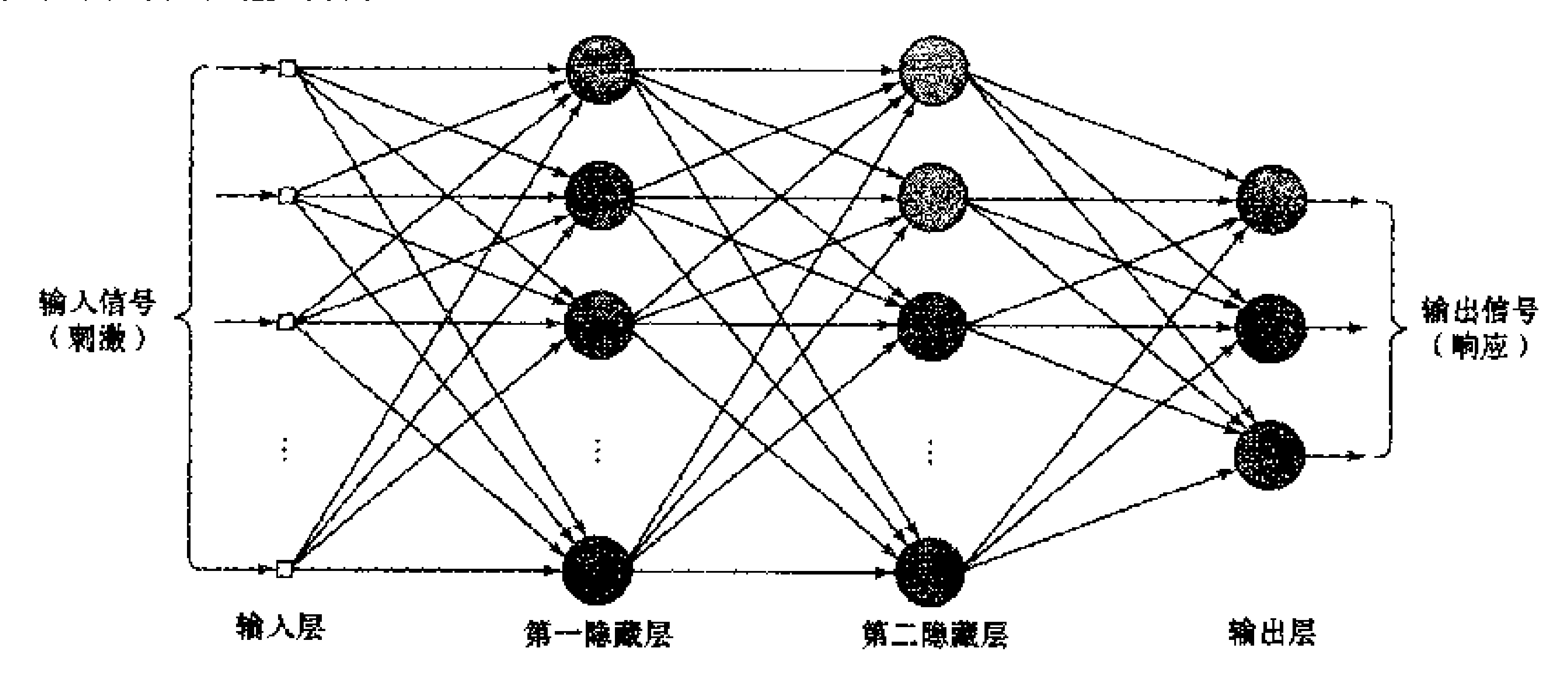


图 4-1 具有两个隐藏层的多层感知器结构图

图 4-2 描绘多层感知器的一部分。两种信号都在这个网络中得到辨认(Parker,1987)：

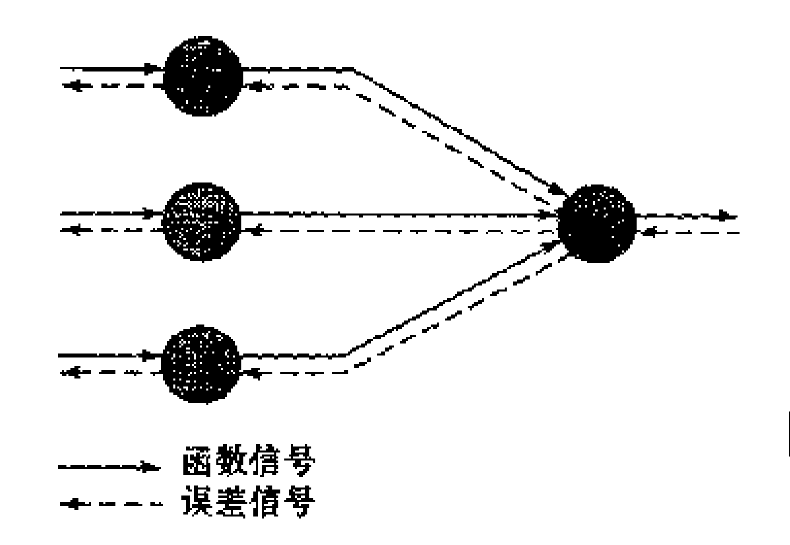


图 4-2 多层感知器中两个基本信号流的方向图示：函数信号的前向传播和误差信号的反向传播

1. 函数信号。一个函数信号是从网络输入层的末端而来的一个输入信号(刺激)，通过网络(一个神经元接一个神经元)传播，到达网络输出层的末端即成为一个输出信号。我们把这样一个信号称之为"函数信号"有两个原因。首先，在网络输出端时假设它表现为有用的函数。第二，在函数信号通过网络上每一个神经元处，该处信号都被当成输入以及与该神经元有关的权值的一个函数来计算的。函数信号也被认为是输入信号。

159

2. 误差信号。一个误差信号产生于网络的一个输出神经元，并通过网络(一层接一层)反向传播。我们称之为"误差信号"是因为网络的每一个神经元对它的计算都以这种或那种形式涉及误差依赖函数。

输出神经元(计算节点)构成网络的输出层，余下的神经元(计算节点)构成网络的隐藏层。因此隐藏层单元并不是网络输出或输入层的一部分——因此它们被称为"隐藏"。第一隐藏层的信号是从由感知单元(源节点)组成输入层馈给的；而它的结果信号又应用于下一个隐藏层；网络的其余部分依此类推。

多层感知器每一个隐藏层或输出层的神经元被设计用来进行两种计算：

1. 计算一个神经元的输出处出现的函数信号，它表现为关于输入信号以及与该神经元

有关的突触权值的一个连续非线性函数。

2. 梯度向量(即误差曲面对与一个神经元输入相连接的权值的梯度)的估计计算，它需要反向通过网络。

反向传播算法的导出是相当地复杂，要减轻这个导出所涉及的数学负担，我们首先给出在推导中使用的符号的一个小结。

符号

- 符号 i，j 和 k 是指网络中不同的神经元；由于信号在网络中从左向右传播，神经元 j 所在层在神经元 i 所在层的右边，而当神经元 j 是隐藏层单元时神经元 k 所在层在神经元 j 所在层的左边。
- 在迭代(时间步) n，网络的第 n 个训练模式(例子)呈现给网络。
- 符号 $\mathscr{E}(n)$ 指迭代 n 时的瞬间误差平方和或瞬间误差能量和。关于所有 n(即整个训练集)的 $\mathscr{E}(n)$ 的平均值即为平均误差能量 $\mathscr{E}_{av}$。
- 符号 $e_j(n)$ 指的是迭代 n 时神经元 j 的输出误差信号。
- 符号 $d_j(n)$ 指的是关于神经元 j 的期望响应并用于计算 $e_j(n)$。
- 符号 $y_j(n)$ 指的是迭代 n 时出现在神经元 j 的输出处的函数信号。
- 符号 $w_{ji}(n)$ 表示突触权值，该权值是迭代 n 时从神经元 i 的输出连接到神经元 j 的输入。这个权值在迭代 n 时的修正量为 $\Delta w_{ji}(n)$。
- 迭代 n 时神经元 j 的诱导局部域(即所有突触输入的加权和加上偏置)记为 $v_j(n)$；它构成作用于神经元 j 激活函数的信号。
- 用来描述神经元 j 的非线性输入——输出函数关系的激活函数表示为 $\varphi_j(\cdot)$。
- 用于神经元 j 的偏置用 b_j 表示；它的作用可由一个与等于 $+1$ 的固定输入相连的权值为 $w_{j0}=b_j$ 突触表示。
- 输入向量(模式)的第 i 个元素用 $x_i(n)$ 表示。
- 输出向量(模式)的第 k 个元素用 $o_k(n)$ 表示。
- 学习率参数记为 η。
- 符号 m_l 表示多层感知器的第 l 层的大小(即节点的数目)；$l=0,1,\cdots,L$，而 L 就是网络的“深度”。因此 m_0 是输入层的大小，m_1 是第一个隐藏层的大小，m_L 是输出层的大小。也使用记号 $m_L=M$。

4.3 反向传播算法

神经元 j 在迭代 n 时(即呈现第 n 个训练例子) 输出误差信号定义如下：

$$e_j(n) = d_j(n) - y_j(n) \qquad \text{神经元 } j \text{ 是输出节点} \tag{4.1}$$

我们将神经元 j 的误差能量瞬间值定义为 $(1/2)e_j^2(n)$。相应的，整个误差能量的瞬间值 $\mathscr{E}(n)$ 即为输出层的所有神经元的误差能量瞬间值的和；这些只是那些误差信号可被直接计算的“可见”神经元。因此，$\mathscr{E}(n)$ 的计算公式是

$$\mathscr{E}(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \tag{4.2}$$

集合 C 包括网络输出层的所有神经元。令 N 记为包含在训练集中模式(例子)的总数。对所有 n 求$\mathscr{E}(n)$的和然后关于集的大小规整化即得的均方误差能量，表示为

$$\mathscr{E}_{av} = \frac{1}{N}\sum_{n=1}^{N}\mathscr{E}(n) \tag{4.3}$$

161

误差能量的瞬间值$\mathscr{E}(n)$和误差能量的平均值$\mathscr{E}_{av}$是网络所有自由参数(即突触权值和偏置水平)的函数。对下一个给定的训练集，$\mathscr{E}_{av}$表示的代价函数作为学习性能的一个量度。学习过程的目的是调整网络的自由参数使得最小化$\mathscr{E}_{av}$。要达到这种最小化，我们使用第 3 章推导 LMS 算法所用原理相似的一个逼近。特别地，我们考虑一个训练的简单方法，即权值在一个模式接一个模式的基础更新，直到一个回合(epoch)结束，也就是整个训练集的完全表示已被网络处理。权值的调整根据每个呈现给网络的模式所计算的各自的误差进行。因此，这些单个权值在训练集上的改变的算术平均，是基于使整个训练集的代价函数$\mathscr{E}_{av}$最小化的真实权值改变的一种估计。在这一节的后面，我们将给出这种估计的性质。

然后考虑图 4-3，它描绘神经元 j 被它左边的一层神经元产生的一组函数信号所馈给。因此，在神经元 j 的激活函数输入处产生的诱导局部域 $v_j(n)$是

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n)y_i(n) \tag{4.4}$$

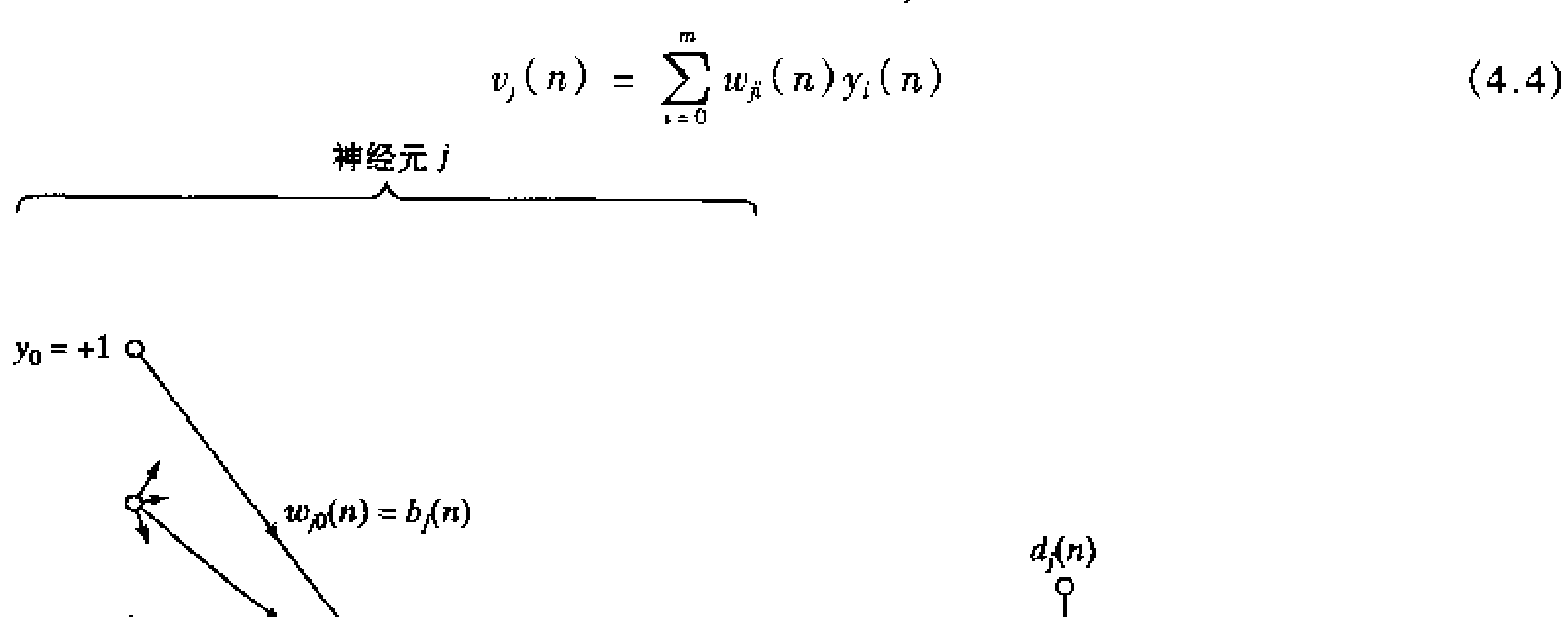


图 4-3 显现输出神经元 j 细节的信号流图

这里 m 是作用于神经元 j 的所有输入(不包括偏置)个数。突触权值 w_{j0}(相应于固定输入 $y_0 = +1$)等于神经元 j 的偏置 b_j。所以迭代 n 时出现在神经元 j 输出处的函数信号 $y_j(n)$是

$$y_j(n) = \varphi_j(v_j(n)) \tag{4.5}$$

反向传播算法以与 LMS 算法类似的方式对突触权值 $w_{ji}(n)$应用一个修正值 $\Delta w_{ji}(n)$，它

正比于$\mathscr{E}(n)$对 $w_{ji}(n)$的偏导数$\partial\mathscr{E}(n)/\partial w_{ji}(n)$。根据微分的链式规则，可以将这个梯度表示为

162

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathscr{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{4.6}$$

偏导数$\partial\mathscr{E}(n)/\partial w_{ji}(n)$代表一个敏感因子，决定突触权值 $w_{ji}(n)$在权值空间的搜索方向。

在式(4.2)两边对 $e_j(n)$取微分，我们得到

$$\frac{\partial \mathscr{E}(n)}{\partial e_j(n)} = e_j(n) \tag{4.7}$$

在式(4.1)两边对 $y_j(n)$取微分，得到

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{4.8}$$

接着，在式(4.5)两边对 $v_j(n)$取微分，得到

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi'_j(v_j(n)) \tag{4.9}$$

最后，在式(4.4)两边对 $w_{ji}(n)$取微分，得到

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n) \tag{4.10}$$

将式(4.7)至(4.10)代入式(4.6)，得到

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi'_j(v_j(n))y_i(n) \tag{4.11}$$

应用于 $w_{ji}(n)$的修正 $\Delta w_{ji}(n)$由 delta 法则定义为

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} \tag{4.12}$$

其中 η 是反向传播算法的学习率参数。式(4.12)中负号的使用意味着在权空间中梯度下降(即寻找一个使得$\mathscr{E}(n)$值下降的权值改变的方向)。于是将(4.11)代入(4.12)中得到

$$\Delta w_{ji}(n) = \eta\delta_j(n)y_i(n) \tag{4.13}$$

这里局域梯度 $\delta_j(n)$定义为

$$\delta_j(n) = -\frac{\partial \mathscr{E}}{\partial v_j(n)} = -\frac{\partial \mathscr{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = e_j(n)\varphi'_j(v_j(n)) \tag{4.14}$$

163

局域梯度指明突触权值所需要的变化。根据(4.14)，输出神经元 j 的局域梯度 $\delta_j(n)$等于该神经元相应误差信号 $e_j(n)$和相应激活函数的导数 $\varphi'_j(v_j(n))$的乘积。

从式(4.13)和(4.14)我们注意到，权值调整 $\Delta w_{ji}(n)$计算所涉及的一个关键因子是神经元 j 输出端的误差信号 $e_j(n)$。在这种情况下，我们要根据神经元的不同位置，区别两种不同的情况。第一种情况，神经元 j 是输出节点。这种情况的处理很简单，因为网络的每一个输出节点都提供自己期望的反应信号，使得计算误差信号成为直截了当的事。在第二种情况，神经元 j 是隐藏层节点。虽然隐藏层神经元不能直接访问，但是它们对网络输出的误差共同承担责任。然而，问题是要知道对隐藏层神经元这种共担的责任如何进行惩罚或奖赏。这就是在 2.7 节中讨论过的信任赋值问题。这已被经过网络反向传播误差信号成功地解决了。

情况 1　神经元 j 是输出节点

当神经元 j 位于网络的输出层时，给它提供自己的一个期望响应。我们可以用式(4.1)来计算这个神经元的误差信号 $e_j(n)$；参看图 4-3。当 $e_j(n)$确定以后，用式(4.14)来计算局域梯度 $\delta_j(n)$是很直接的。

情况 2　神经元 j 是隐藏层节点

当神经元 j 位于网络的隐藏层时，就没有对该输入神经元的指定期望响应。因此，隐藏层的误差信号要根据所有与隐藏层神经元直接相连的神经元的误差来递归决定。这就是为什么反向传播算法的发展变得很复杂的地方。考虑在图 4-4 中所描绘的情况，它描绘的神经元 j 就是一个网络隐藏层节点。根据式(4.14)我们可把隐藏层神经元的局域梯度重新定义为

$$\delta_j(n) = -\frac{\partial \mathscr{E}(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial \mathscr{E}(n)}{\partial y_j(n)}\varphi'_j(v_j(n)), \qquad \text{神经元 } j \text{ 是隐藏的} \tag{4.15}$$

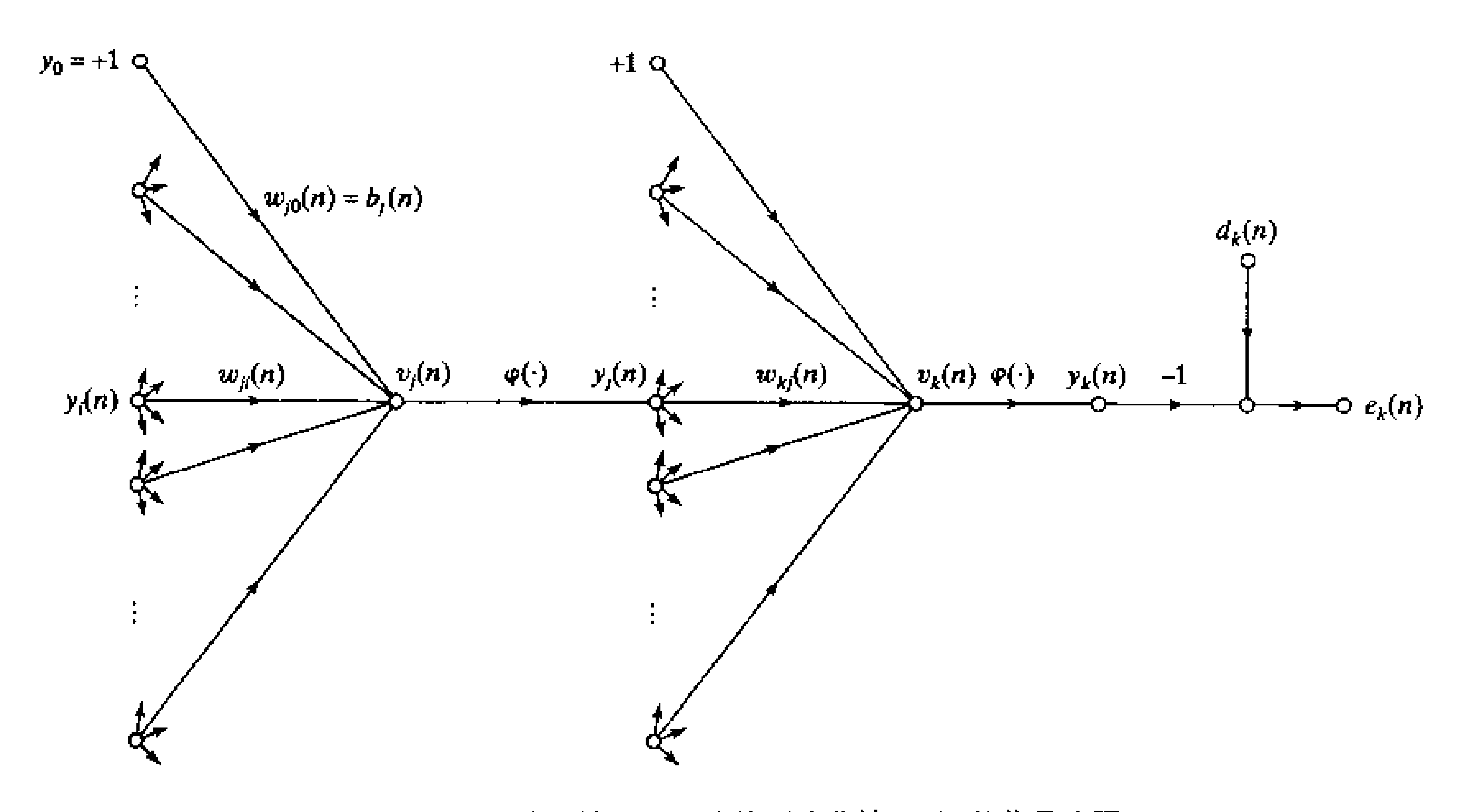


图 4-4　显现输出神经元 k 连接到隐藏神经元 j 的信号流图

在公式的第二行我们用到了式(4.9)。要计算偏导$\partial\mathscr{E}(n)/\partial y_j(n)$我们进行如下处理。从图4-4可以看到

$$\mathscr{E}(n) = \frac{1}{2}\sum_{k\in C} e_k^2(n), \qquad \text{神经元 } k \text{ 是输出节点} \tag{4.16}$$

这就是对式(4.2)用下标 k 替代下标j。我们这么写是为了避免与在情况 2 使用下标 j 表示一个隐藏神经元相混淆。在式(4.16)两边对函数信号 $y_j(n)$求偏导，得到

164

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)} \tag{4.17}$$

接着我们对偏导数$\partial e_k(n)/\partial y_j(n)$使用链式规则，重写式(4.17)为等价形式

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)} \tag{4.18}$$

然而，从图4-4我们注意到

$$e_k(n) = d_k(n) - y_k(n) = d_k(n) - \varphi_k(v_k(n)), \qquad \text{神经元 } k \text{ 为输出节点} \tag{4.19}$$

因此

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi'_k(v_k(n)) \tag{4.20}$$

我们从图4-4也要注意到对神经元k来说，诱导局部域是

$$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n) \tag{4.21}$$

165 这里m是神经元k所有输入的个数(不包括偏置)。同样在这里突触权值$w_{k0}(n)$等于应用于神经元k的偏置$b_k(n)$，相应的输入是固定在值+1处的。求(4.21)对$y_j(n)$的微分得到

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n) \tag{4.22}$$

用式(4.20)和(4.22)代入(4.18)，我们得到期望的偏微分

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = -\sum_k e_k(n)\varphi'_k(v_k(n)) w_{kj}(n) = -\sum_k \delta_k(n) w_{kj}(n) \tag{4.23}$$

在第二行用到局域梯度$\delta_k(n)$的定义，它由式(4.14)给出，其中用下标k替代j。

最后，用式(4.23)代入(4.15)，得到关于局域梯度$\delta_j(n)$的反向传播公式

$$\delta_j(n) = \varphi'_j(v_j(n)) \sum_k \delta_k(n) w_{kj}(n), \qquad \text{神经元 } j \text{ 为隐藏单元} \tag{4.24}$$

图4-5代表式(4.24)的信号流图，假设输出层有m_L个神经元。

在式(4.24)中与局域梯度$\delta_j(n)$的计算有关的因子$\varphi'_j(v_j(n))$仅仅依赖于隐藏层神经元j的激活函数。这个计算涉及的其余因子，也就是所有神经元k的和，依赖于两组项。第一组项$\delta_k(n)$，对于紧接隐藏层神经元j右端的层中直接与神经元j相连的所有神经元，需要具有误差信号$e_k(n)$的知识：参看图4-4。第二组项$w_{kj}(n)$是由所有这些连接的突触权值组成的。

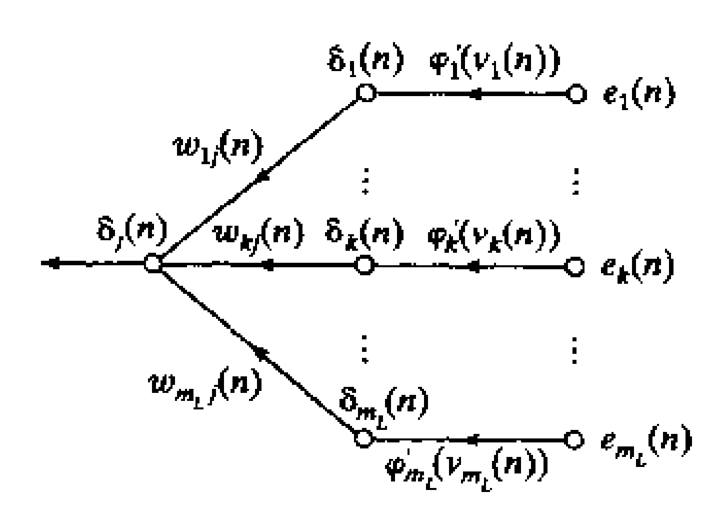


图4-5 误差信号反向转播伴随系统的部分信号流图

现在，我们总结为反向传播算法导出的关系。首先，由神经元i连接到神经元j的突触权值的校正值$\Delta w_{ji}(n)$由delta规则定义如下：

166

$$\begin{pmatrix} \text{权值} \\ \text{校正} \\ \Delta w_{ji}(n) \end{pmatrix} = \begin{pmatrix} \text{学习率} \\ \text{参数} \\ \eta \end{pmatrix} \cdot \begin{pmatrix} \text{局部} \\ \text{梯度} \\ \delta_j(n) \end{pmatrix} \cdot \begin{pmatrix} \text{神经元 } j \\ \text{输入信号} \\ y_i(n) \end{pmatrix} \tag{4.25}$$

其次，局域梯度$\delta_j(n)$取决于神经元j是一个输出节点还是一个隐藏层节点：

1. 如果神经元 j 是一个输出节点，$\delta_j(n)$等于导数 $\varphi'_j(v_j(n))$和误差信号 $e_j(n)$的乘积，它们都和神经元 j 相关联；参看式(4.14)。

2. 如果神经元 j 是隐藏层节点，$\delta_j(n)$等于相应导数 $\varphi'_j(v_j(n))$和 δ_j 的加权和的乘积，这些 δ_j 是对与神经元 j 相连的下一个隐藏层或输出层中的神经元计算得到的；参看式(4.24)。

计算的两次通过

在反向传播算法的应用中，计算有两种截然不同的通过。第一个通过是指前向通过，而第二个是指反向通过。

在前向通过中，经过网络时突触权值保持不变，而网络的函数信号在一个神经元接一个神经元基础上计算。出现在神经元 j 输出处的函数信号计算为

$$y_j(n) = \varphi(v_j(n)) \tag{4.26}$$

其中 $v_j(n)$是神经元 j 的诱导局部域，由

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n)y_i(n) \tag{4.27}$$

定义，这里，m 是神经元 j 的所有输入的数量(不包括偏置)，而 $w_{ji}(n)$是连接神经元 i 和神经元 j 的突触权值，$y_i(n)$是指神经元 j 的输入信号或是出现在神经元 i 的输出端的函数信号。如果神经元 j 在网络的第一隐藏层，则 $m = m_0$ 且下标 i 是指网络的第 i 个输入端点，我们写作

$$y_i(n) = x_i(n) \tag{4.28}$$

这里 $x_i(n)$是指输入向量(模式)的第 i 个元素。在另一方面，如果神经元 j 在网络的输出层，则 $m = m_L$，并且下标 j 是指网络的第 j 个输出端点，我们写作

$$y_j(n) = o_j(n) \tag{4.29}$$

167

这里 $o_j(n)$是指输出向量(模式)的第 j 个元素。这个输出和期望响应 $d_j(n)$相比较，得到第 j 个输出神经元的误差信号。因此，计算的前向阶段由输入向量馈给的第一个隐藏层开始，以输出层计算该层的每一个神经元的误差信号而结束。

在另一方面，反向通过从输出层开始，误差信号向左经过网络一层一层传播，并且递归计算每一个神经元的 δ(即局部梯度)。该递归过程允许突触权值根据式(4.25)的 delta 规则变化。对于位于输出层的神经元，δ 简单地等于这个神经元的误差信号乘以它的非线性一次导数。因此，我们使用式(4.25)来计算所有馈入输出层的连接的权值变化。给出输出层神经元的 δ，接着用式(4.24)来计算倒数第二层的所有神经元的 δ 和所有馈入该层的连接的权值变化。通过传播这个变化给网络的所有突触权值，一层接一层连续递归计算。

注意由于每给出一个训练例子，其输入模式在整个往返过程中是固定的(钳制的)，这个往返过程包括前向通过和随后的反向通过。

激活函数

计算多层感知器每一个神经元的 δ 需要关于神经元的激活函数 $\varphi(\cdot)$的导数知识。要导数存在，则需要函数 $\varphi(\cdot)$连续。用基本术语，激活函数必需满足的要求是可微性。通常用于多层感知器的连续可微非线性激活函数的一个例子是 sigmoid 非线性性；这里有两种形式要说一下：

1. logistic 函数。这种 sigmoid 非线性性的一般形式由

$$\varphi_j(v_j(n)) = \frac{1}{1+\exp(-av_j(n))} \qquad a>0, -\infty < v_j(n) < \infty \tag{4.30}$$

定义，这里 $v_j(n)$是神经元 j 的诱导局部域。根据这种非线性性，输出的范围位于 $0 \leq y_j \leq 1$ 之内。对式(4.30)取 $v_j(n)$的微分，我们得到

$$\varphi'_j(v_j(n)) = \frac{a\exp(-av_j(n))}{[1+\exp(-av_j(n))]^2} \tag{4.31}$$

由于 $y_j(n)=\varphi_j(v_j(n))$，我们可以从式(4.31)中消去指数项 $\exp(-av_j(n))$，所以导数 $\varphi'_j(v_j(n))$可以表示为

$$\varphi'_j(v_j(n)) = ay_j(n)[1-y_j(n)] \tag{4.32}$$

因为神经元 j 位于输出层，所以 $y_j(n)=o_j(n)$。因此可以将神经元 j 的局域梯度表示为

$$\delta_j(n) = e_j(n)\varphi'_j(v_j(n)) = a[d_j(n)-o_j(n)]o_j(n)[1-o_j(n)] \tag{4.33}$$

这里的 $o_j(n)$是神经元 j 输出端的函数信号，而 $d_j(n)$是它的期望响应。另一方面，对任意的一个隐藏层神经元 j，我们可以将局域梯度表示为

$$\begin{aligned}\delta_j(n) &= \varphi'_j(v_j(n))\sum_k \delta_k(n)w_{kj}(n)\\ &= ay_j(n)[1-y_j(n)]\sum_k \delta_k(n)w_{kj}(n), \qquad j \text{ 为隐藏神经元}\end{aligned} \tag{4.34}$$

168

从式(4.32)可以看出，导数 $\varphi'_j(v_j(n))$当 $y_j(n)=0.5$ 时取最大值，当 $y_j(n)=0$ 或 $y_j(n)=1$ 时取它的最小值(0)。既然网络的一个突触权值的变化总量与导数 $\varphi_j(v_j(n))$成比例，因此对于一个 sigmoid 激活函数来说，突触权值改变最多的神经元是那些函数信号在它们的中间范围之内的网络的神经元。根据 Rumelhart et al.(1986a)，正是反向传播学习这个特点导致它作为学习算法的稳定性。

2. 双曲正切函数。另外一个经常使用的 sigmoid 非线性形式是双曲正切函数，它的最通用的形式由

$$\varphi_j(v_j(n)) = a\tanh(bv_j(n)), \qquad (a,b)>0 \tag{4.35}$$

定义，这里 a 和 b 是常数。事实上，双曲正切函数只是伸缩和平移的 logistic 函数。它对 $v_j(n)$的导数如下：

$$\begin{aligned}\varphi'_j(v_j(n)) &= ab\,\text{sech}^2(bv_j(n)) = ab(1-\tanh^2(bv_j(n)))\\ &= \frac{b}{a}[a-y_j(n)][a+y_j(n)]\end{aligned} \tag{4.36}$$

如果神经元 j 位于输出层，它的局域梯度是

$$\delta_j(n) = e_j(n)\varphi'_j(v_j(n)) = \frac{b}{a}[d_j(n)-o_j(n)][a-o_j(n)][a+o_j(n)] \tag{4.37}$$

如果神经元 j 位于隐藏层，我们有

$$\begin{aligned}\delta_j(n) &= \varphi'_j(v_j(n))\sum_k \delta_k(n)w_{kj}(n)\\ &= \frac{b}{a}[a-y_j(n)][a+y_j(n)]\sum_k \delta_k(n)w_{kj}(n), \qquad j \text{ 为隐藏神经元}\end{aligned} \tag{4.38}$$

对 logistic 函数使用式(4.33)和(4.34)以及对双曲正切函数使用式(4.37)和(4.38)，我们不需

要激活函数的具体信息就可以计算局域梯度 δ_j。

学习率

反向传播算法提供使用最速下降方法在权空间计算得到的轨迹的一种近似。我们使用的学习率参数 η 越小，从一次迭代到下一次迭代的网络突触权值的变化量就越小，轨迹在权值空间就越光滑。然而，这种改进是以减慢学习速度为代价的。另一方面，如果我们让 η 的值太大以加快学习速度的话，结果就有可能使网络的突触权值的变化量不稳定(即振荡)。一个既要加快学习速度又要保持稳定的简单方法是修改式(4.13)的 delta 法则，使它包括动量项[2]，表示为(Rumelhart et al.,1986a)

169

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \tag{4.39}$$

这里 α 是动量常数，通常是正数。它控制围绕 $\Delta w_{ji}(n)$ 反馈环路，如图 4-6 所示，其中 z^{-1} 表示单位延迟操作符。式(4.39)被称之为广义 delta 规则[3]；它包括式(4.13)的 delta 规则的作为特殊情况(即 $\alpha = 0$)。

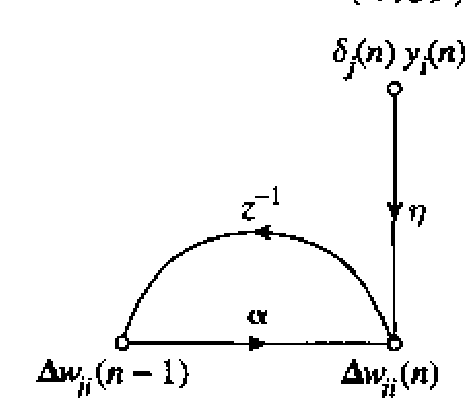


图 4-6 说明动量常数 α 作用的信号流图

为了看出由于动量函数 α 在一系列模式呈现上对突触权值的影响，我们将式(4.39)重新写为带下标 t 的一个时间序列。索引 t 从初始时刻 0 到当前时刻 n。式(4.39)可被视为权值修正量 $\Delta w_{ji}(n)$ 的一阶差分方程。解这个关于 $\Delta w_{ji}(n)$ 的方程得到

$$\Delta w_{ji}(n) = \eta \sum_{t=0}^{n} \alpha^{n-t} \delta_j(t) y_i(t) \tag{4.40}$$

这代表一个长度为 $n+1$ 的时间序列。从式(4.11)和(4.14)，我们可知 $\delta_j(n) y_j(n)$ 等于 $-\partial \mathscr{E}(n)/\partial w_{ji}(n)$。因此我们将方程(4.40)重写为等价形式

$$\Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} \alpha^{n-t} \frac{\partial \mathscr{E}(t)}{\partial w_{ji}(t)} \tag{4.41}$$

在这个关系的基础上，我们来做以下深入观察(Watrous,1987;Jacobs,1988)：

1. 当前修正值 $\Delta w_{ji}(n)$ 代表指数加权的时间序列的和。欲使时间序列收敛，动量常数必须限制在 $0 \leqslant |\alpha| < 1$ 范围内。当 α 等于 0 时，反向传播算法运行起来没有动量。虽然在实际中动量常数 α 不大可能是负的，但它还是可正可负。

2. 当偏导数 $\partial \mathscr{E}(t)/\partial w_{ji}(t)$ 在连续迭代中有相同的代数符号，指数加权和 $\Delta w_{ji}(n)$ 在数量增加，所以，权值 $w_{ji}(n)$ 被大幅度调整。在反向传播算法中包含动量趋于在稳定的下降方向上加速下降。

3. 当偏导数 $\partial \mathscr{E}(t)/\partial w_{ji}(t)$ 在连续迭代中有相反的代数符号，指数加权和 $\Delta w_{ji}(n)$ 在数量上减少，所以，权值 $w_{ji}(n)$ 调整不大。在反向传播算法中包含动量具有稳定符号正负摆动方向的效果。

170

在反向传播算法中，动量的使用对更新权值来说的一个较小的变化，而它对算法的学习可能会有一些有利的影响。动量项对于使学习过程不停止在误差曲面上一个浅层的局部最小可能也有益处。

在导出反向传播算法时假设学习率参数 η 是一个常数。然而，事实上它应该被定义为

η_{ji}；也就是说，学习率参数应该是依赖连接的。确实，在网络的不同地方而使用不同的学习率参数会发生很多有趣的事情。关于这一点在下一节我们会给出详细描述。

同样值得注意的是，我们在反向传播算法的应用中可以选择所有突触权值都是可调整的，或者在自适应过程中可能限制网络中某些权值使其保持固定。对于后者，误差信号是以通常的方式通过网络反向传播的；然而，固定的突触权值是不更改的。这一点，可以简单通过使突触权值的学习率参数 η_{ji} 等于0来做到。

训练的串行和集中方式

在反向传播算法的实际应用中，学习结果是从将指定的训练例子多次呈现给多层感知器而得到的。像前面提到过的一样，在一个学习过程中整个训练集的完全呈现称之为一个回合(epoch)。学习过程是在一个回合接一个回合的基础上进行直到网络的突触权值和误差水平稳定下来，并且整个训练集上的均方误差收敛于某个极小值。从一个回合到下一个回合时将训练样本的呈现顺序随机化是一个很好的实践。这种随机化易于在学习循环中使得权空间搜索具有随机性，因此可以在突触权值向量演化中避免极限环出现的可能性；极限环在第14章讨论。

对于一个给定的训练集，反向传播学习可能会以下面两种基本方式中的一种进行：

1. 串行方式。反向传播学习的串行方式也称为是在线方式、模式方式或随机方式。在这种运行方式里在每个训练样本呈现之后进行权值更新；这正是导出目前反向传播算法公式所引用的运行方式。具体地，考虑包含 N 个训练例子(模式)的一个回合，其顺序是$(\mathbf{x}(1),\mathbf{d}(1)),\cdots,(\mathbf{x}(N),\mathbf{d}(N))$。该回合的第一个例子对$(\mathbf{x}(1),\mathbf{d}(1))$呈现给网络时，完成以前描述的前向和反向计算顺序，导致网络的突触权值和偏置水平的一定调整。接着，该回合的第二个样本对$(\mathbf{x}(2),\mathbf{d}(2))$呈现时，重复前向和反向的计算顺序，导致网络的突触权值和偏置水平的进一步调整。直到该回合的最后一个例子对$(\mathbf{x}(N),\mathbf{d}(N))$考虑完以后这个过程才结束。 [171]

2. 集中方式。在反向传播学习的集中方式中，权值更新要在组成一个回合的所有训练例子呈现后才进行。对于特定的一个回合，我们将代价函数定义为式(4.2)和(4.3)均方误差，这里重新写成组合形式

$$\mathscr{E}_{av}=\frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C}e_j^2(n)\tag{4.42}$$

这里误差信号 $e_j(n)$表示训练例子 n 由式(4.1)中所定义的输出神经元 j 有关的误差。误差 $e_j(n)$等于 $d_j(n)$和 $y_j(n)$的差，它们分别表示期望响应向量 $\mathbf{d}(n)$的第 j 个分量和网络输出的相应值。在式(4.42)中关于 j 的内层求和是对网络的输出层的所有神经元进行的，而关于 n 的外层求和是对当前回合的整个训练集进行的。对于学习率参数 η，应用于从 i 连接到 j 的 w_{ji}的修正值由 delta 规则

$$\Delta w_{ji}=-\eta\frac{\partial\mathscr{E}_{av}}{\partial w_{ji}}=-\frac{\eta}{N}\sum_{n=1}^{N}e_j(n)\frac{\partial e_j(n)}{\partial w_{ji}}\tag{4.43}$$

定义。要计算偏导数$\partial e_j(n)/\partial w_{ji}$，我们用以前的相同方式处理。根据式(4.43)，在集中方式中，权值的校正值 Δw_{ji}是在整个训练集提交训练以后才决定。

从在线运行的观点来看，训练的串行方式比集中方式要好，因为对每一个突触权值来说需有更少的局部存储。而且，既然以随机方式给定网络的训练模式，利用一个模式接一个模式的方法更新权值使得在权值空间的搜索自然具有随机性。这使得反向传播算法陷入局部最

小的可能性降低了。

同样地，串行方式的随机性质使得要得到算法收敛的理论条件变得困难了。比较而言，训练集中方式的使用为梯度向量提供了一个精确的估计；收敛到局部最小只要简单的条件就可以保证。集中方式的成分比串行方式更容易并行化。

当训练数据冗余时(即数据集合包含同一模式的几个备份)，我们发现不像集中方式那样，因为在一次只呈现一个例子，从而串行方式可以利用这种冗余。当数据集很大且高度冗余时尤其如此。

总地来说，尽管反向传播学习的串行方式有一些缺点，但它能够如此流行(特别对解决模式分类问题)有两个重要的原因：

- 算法的实现很简单。
- 它为大型问题和困难的问题提供有效的解决方法。

172

停止准则

通常，不能证明反向传播算法收敛，并且没有明确定义的停止它运行的准则。相反，仅有一些合理的准则，它们每个都有自己的实际用处，这些准则可以用于终止权值的调整。要提出这样一个准则，考虑关于误差曲面的局部或全局最小的特殊性质是符合逻辑的。将权值向量 $\mathbf{w}^*$ 标记为局部或全局最小点。要使 $\mathbf{w}^*$ 成为最小点的一个必要条件是误差曲面对权值向量 $\mathbf{w}$ 的梯度向量 $\mathbf{g}(\mathbf{w})$(即一阶偏导数)在 $\mathbf{w}=\mathbf{w}^*$ 处等于 $\mathbf{0}$。因此，我们可以提出反向传播学习的一个合理的收敛准则(Kramer and Sangiovanni-Vincentelli, 1989)：

当梯度向量的欧几里德范数达到一个充分小的梯度阈值时，认为反向传播算法已经收敛。

这个收敛准则的缺点是，为了成功试验，学习时间可能会很长。同时它需要计算梯度向量 $\mathbf{g}(\mathbf{w})$。

另一个我们能够使用的最小点的特殊性质是代价函数或误差量度 $\mathscr{E}_{av}(\mathbf{w})$ 在 $\mathbf{w}=\mathbf{w}^*$ 处是平稳的。因此，我们可以建议一个不同的收敛准则：

当每一个回合的均方误差的变化的绝对速率足够小时，认为反向传播算法已经收敛。

均方误差的变化的绝对速率如果每个回合是在百分之 0.1 到 1 之间，一般认为它足够小。有时候，每一个回合都会用到小到百分之 0.01 这样的值。不幸的是，这个准则可能会导致学习过程的过早终止。

有另外一个有用的且有理论支持的收敛准则。在每一个学习迭代之后，都要检查网络的泛化性能。当泛化性能是适当的，或泛化性能明显达到峰值时，学习过程被终止：参看 4.14 节有更多细节。

4.4 反向传播算法小结

图 4-1 给出一个多层感知器的结构布局。反向传播学习的相应的信号流图，包括学习过程计算的前向和反向阶段，$L=2$ 和 $m_0=m_1=m_2=3$ 的情况在图 4-7 中表示。信号流图的上面一部分是说明前向通过的。信号流图的下面一部分是说明反向通过的，这也称为在反向传播算法中计算局域梯度的灵敏图(sensitivity graph)(Narendra and Parthasarathy, 1990)。

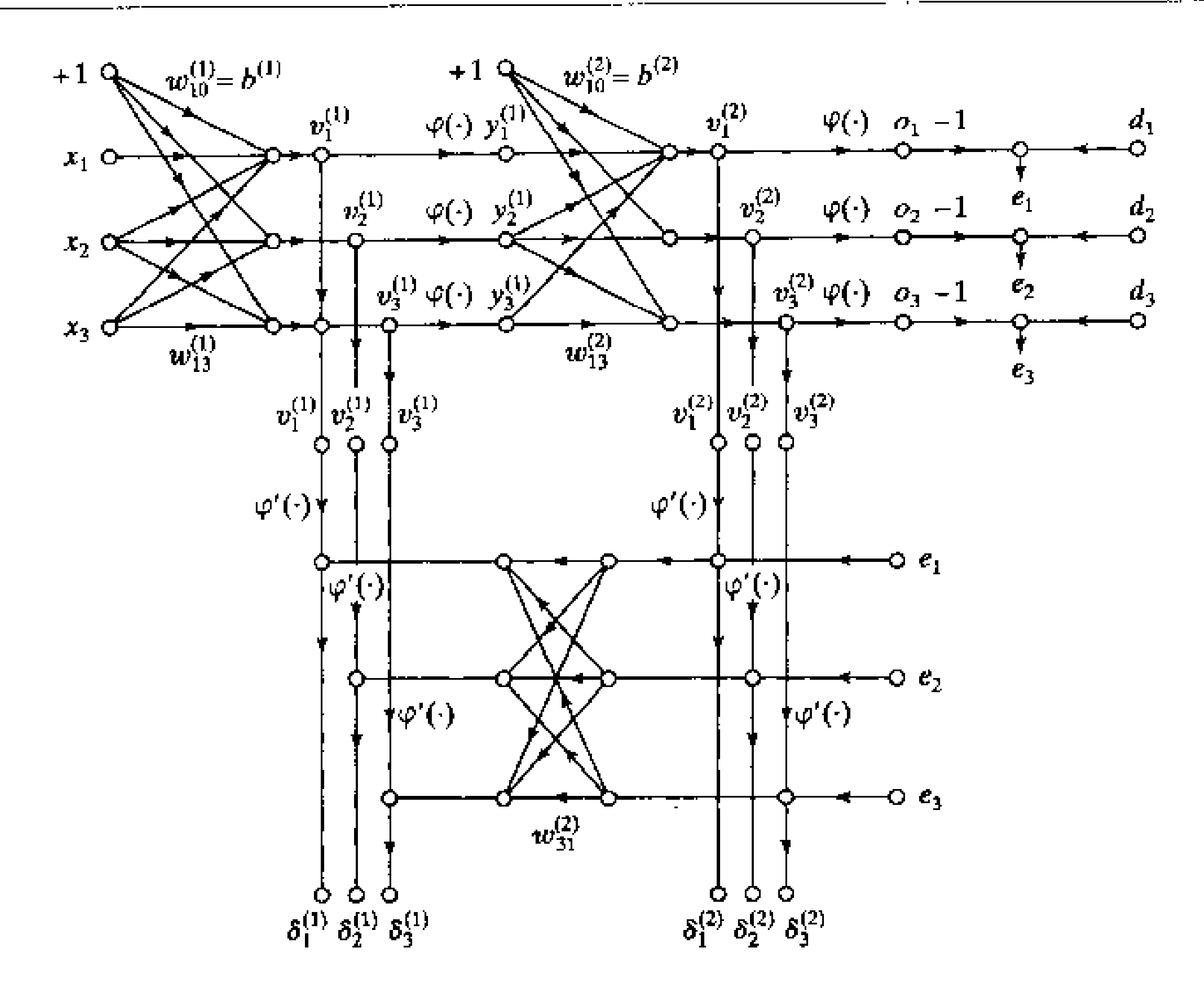


图 4-7 反向传播学习信号流图小结
图顶部：前向通过 图底部：反向通过

前面我们提到权值的串行更新是反向传播算法的在线实现的更好方法。对这种方式运行，算法通过训练样本$\{(\mathbf{x}(n),\mathbf{d}(n))\}_{n=1}^{N}$进行循环如下：

1. 初始化。假设没有先验知识可用，我们以一个随机分布随机地挑选突触权值和阈值，这个分布选择为均值等于0的均匀分布，它的方差的选择应该使得神经元的诱导局部域的标准偏差位于 sigmoid 激活函数的线形部分与饱和部分过渡处。

173

2. 训练样本的呈现。呈现训练样本的一个回合给网络。对训练集中以某种形式排序的每个样本，依次进行在下面的第3点和第4点中所描述的前向和反向计算。

3. 前向计算。在该回合中设一个训练样本是$(\mathbf{x}(n),\mathbf{d}(n))$，输入向量$\mathbf{x}(n)$指向感知节点的输入层和期望响应向量$\mathbf{d}(n)$指向计算节点的输出层。不断地经由网络一层一层地前进，可以计算网络的诱导局部域和函数信号。在层 l 的神经元j 的诱导局部域$v_j^{(l)}(n)$为

$$v_j^{(l)}(n) = \sum_{i=0}^{m_0} w_{ji}^{(l)}(n)y_i^{(l-1)}(n) \tag{4.44}$$

这里$y_i^{(l-1)}(n)$是迭代 n 时前面第 $l-1$ 层的神经元 i 的输出(函数)信号，而 $w_{ji}^{(l)}(n)$是从第 $l-1$层的神经元 i 指向第 l 层的神经元j 的权值。对 $i=0$，我们有 $y_0^{(l-1)}(n)=+1$，并且 $w_{j0}^{(l)}(n)=b_j^{(l)}(n)$是第 l 层的神经元j 的偏置。假设使用一个 sigmoid 函数，则第 l 层的神经元j 的输出信号是

174

$$y_j^{(l)} = \varphi_j(v_j(n))$$

如果神经元 j 是在第一隐藏层(即 $l=1$)，置

$$y_j^{(0)}(n) = x_j(n)$$

这里 $x_j(n)$是输入向量 $\mathbf{x}(n)$的第 j 个元素。如果神经元 j 在输出层(即 $l=L$，这里的 L 称为网络的深度)，令

$$y_j^{(L)} = o_j(n)$$

计算误差信号

$$e_j(n) = d_j(n) - o_j(n) \tag{4.45}$$

这里 $d_j(n)$是期望响应向量 $\mathbf{d}(n)$的第 j 个向量。

4. 反向计算。计算网络的 δ(即局域梯度)，定义为

$$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n)\varphi'_j(v_j^{(L)}(n)) & \text{对输出层 } L \text{ 的神经元} j \\ \varphi'_j(v_j^{(l)}(n)) \sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n) & \text{对隐藏层 } l \text{ 的神经元} j \end{cases} \tag{4.46}$$

这里 $\varphi'_j(\cdot)$是指对自变量的微分。根据广义 delta 规则调节网络第 l 层的突触权值：

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n-1)] + \eta\delta_j^{(l)}(n)y_i^{(l-1)}(n) \tag{4.47}$$

这里 η 为学习率参数，α 为动量常数。

5. 迭代。通过呈现新的一回合样本给网络根据第 3 点和第 4 点进行前向和反向迭代计算，直到满足停止准则。

注意：训练样本的呈现顺序从一个回合到另一个回合必须是随机的。动量和学习率参数随着训练迭代次数的增加而调整(通常是减少的)。以后会给出这些点的理由。

4.5 异或问题

一个基本的(单层)感知器没有隐藏神经元。因此，它不能对非线性可分的输入模式分类。然而，非线性可分模式却是很普遍的。例如，对异或(XOR)问题就遇到这种情形，它可以看作在单位超立方体中更一般的点分类问题的特例。在超立方体中的每个点不是属于类 0 就是属于类 1。但是对异或问题特殊情形，我们仅考虑单位正方形的四个角，相应的输入模式为(0,0)，(0,1)，(1,0)和(1,1)。第一个和第三个输入模式属于类 0，即 [175]

$$0 \oplus 0 = 0$$

和

$$1 \oplus 1 = 0$$

这里⊕指的是异或布尔函数运算符。输入模式(0,0)和(1,1)是单位正方形的两个相对的角，但它们产生相同的结果是 0。另一方面，输入模式(0,1)和(1,0)是单位正方形的另一对相对的角，但是它们属于类 1，即

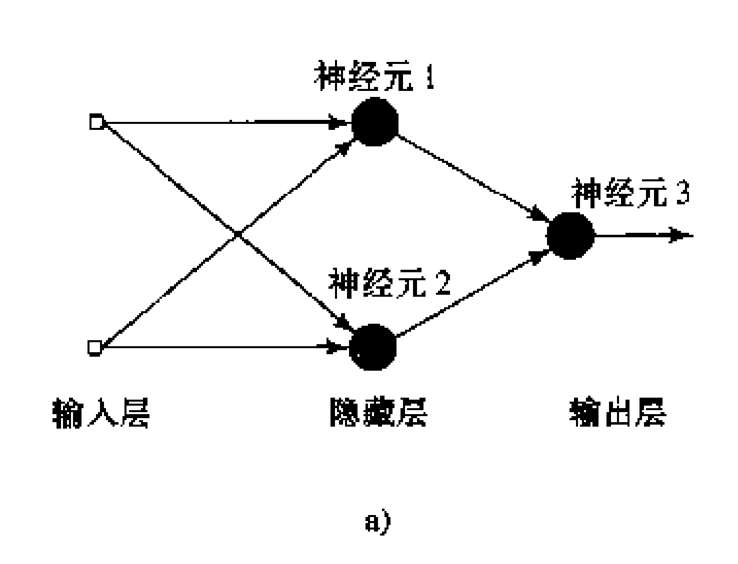
$$0 \oplus 1 = 1$$

和

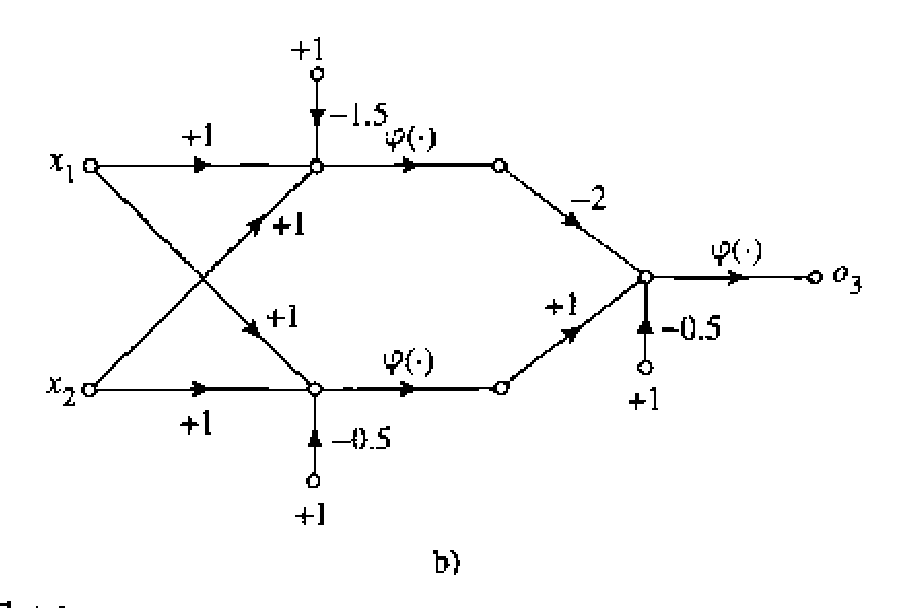
$$1 \oplus 0 = 1$$

首先我们知道有两个输入的单个神经元的使用得到的决策边界是输入空间的一条直线。在这条直线的一边的所有的点，神经元输出 1；而在这条直线的另一边的点，神经元输出 0。在输入空间中这条直线的位置和方向由与两个输入节点相连的神经元的突触权值和它的偏置决定。由于输入模式(0,0)和(1,1)是位于单位正方形的相对的两个角，输入模式(0,1)和(1,0)也一样，很清楚我们作不出这样一条直线作为决策边界可以使(0,0)和(1,1)在一个区域，而(1,0)和(0,1)在另一区域。换句话说，一个简单感知器不能解决 XOR 问题。

如图 4-8a 中所示，我们可以使用一层有两个神经元的隐藏层来解决异或问题(Touretzky and Pomerleau, 1989)。网络的信号流图在图 4-8 b给出。这里作以下假设：



a)



b)

图 4-8

a)解决 XOR 问题的网络结构图 b)网络信号流图

- 每一个神经元都由一个McCulloch-Pitts模型表示，使用阈值函数作为它的激活函数。
- 比特符号 0 和 1 分别由水平 0 和 +1 表示。

隐藏层中顶部神经元标记为 1，定义为

$$w_{11} = w_{12} = +1$$

$$b_1 = -\frac{3}{2}$$

该隐藏神经元构造的决策边界的斜率等于 -1，在图 4-9a给出其位置。在隐藏层中底部神经元标记为 2，定义为

176

$$w_{21} = w_{22} = +1$$

$$b_2 = -\frac{1}{2}$$

第二隐藏神经元构造的决策边界的方向和位置由图4-9b 给出。

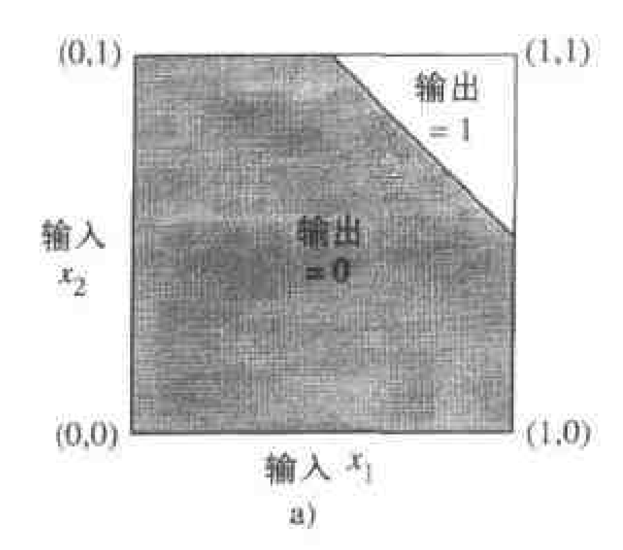
图 4-8a 的标记为 3 的输出神经元定义为

$$w_{31} = -2$$

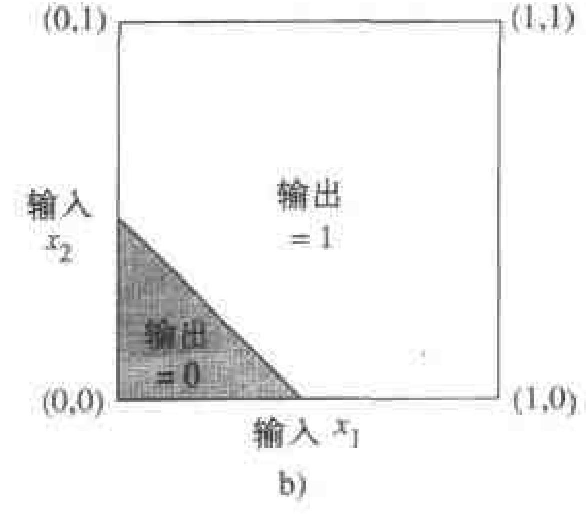
$$w_{32} = +1$$

$$b_3 = -\frac{1}{2}$$

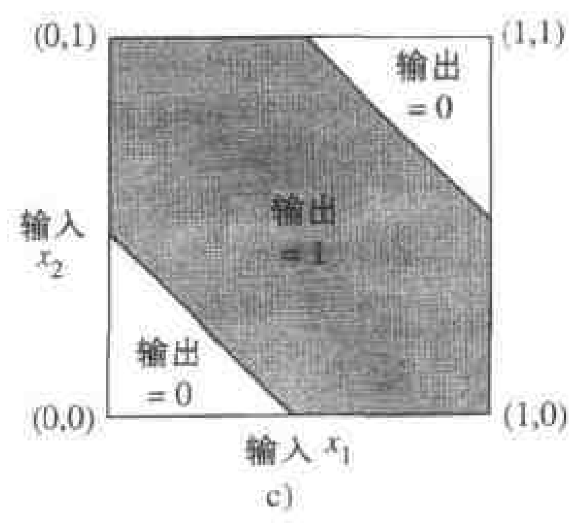
输出神经元的功能是对两个隐藏神经元形成的决策边界构造线性组合。这个计算结果表示在图 4-9c 中。底部隐藏神经元由一个兴奋(正)连接到输出神经元，而顶部隐藏神经元由一个更强的抑制(负)连接到输出神经元。当两个隐藏神经元都断开时，这种情况当输入信号是(0,0)时发生，输出神经元保持断开。当两个隐藏神经元都接通时，这种情况当输入模式是(1,1)时发生，输出神经元也保持断开，因为由连向顶部隐藏神



a)



b)



c)

图 4-9

a)在图 4-8 中的网络隐藏神经元 1 所构造的决策边界 b)网络隐藏神经元 2 所构造的决策边界 c)整个网络所构造的决策边界

经元负权值产生的抑制效果超过由连向底部隐藏神经元正权值产生的兴奋效果。当顶部隐藏神经元是断开的而底部隐藏神经元是接通的，即输入模式是(0,1)或(1,0)时，输出神经元是接通的，因为正的权值连向了底部隐藏神经元。因此图 4-8a 确实解决了异或问题。 177

4.6 改善反向传播算法性能的试探法

人们常说，用于反向传播算法的神经网络的设计与其说是科学，不如说更像一门艺术，因为这个设计中的很多数值因素依赖于个人自己的经验。从某种意义上讲这个论断是正确的。但是，也有些方法能对反向传播算法有重大提高，可描述如下：

1. 串行更新而不是集中方式更新。如前面已经提到过的，反向传播学习的串行方式(涉及一个模式接一个模式的更新)要比集中方式的计算快。特别当训练数据集很大且高度冗余时，更是如此。(高度冗余的数据对集中方式更新所需要的 Jacobi 矩阵的估计提出了计算上的问题。) 178

2. 最大可能的信息内容。作为一个基本的规则，对呈现给反向传播算法的每一个训练样本的挑选必须建立在其信息内容对解决问题有最大可能的基础上(LeCun,1993)。达到这个目标的两种方法是：

- 使用训练误差最大的样本。
- 使用的样本要与以前使用的有根本区别。

这两个试探方法起因于对权空间进行更多搜索的愿望。

在模式分类的任务中使用串行反向传播学习，经常使用的一个简单技巧是将样本每个回合呈现给多层感知器的顺序随机化(即弄乱)。理想情况下，随机化可以确保一个回合中的相继的样本很少属于同一类。

对于一个更加改良的技巧，我们使用强调图表，这涉及呈现给网络更加困难的模式而不是容易的模式。一个特定的模式是容易还是困难可以通过检查其产生的误差与算法以前迭代所产生的误差进行比较来确认。然而，在使用强调图表时有两个问题需要仔细注意：

- 一个回合中呈现给网络的样本分布是变形的。
- 例外点或是错误标记的样本的出现对于算法的性能会有一个灾难性的后果；学习这样的例外点对网络在输入空间中更大可能区域的泛化能力带来损害。

3. 激活函数。一般来说，当网络的神经元模型嵌入的 sigmoid 激活函数是反对称而不是非对称时，一个用反向传播算法训练的多层感知器会学得快一些；详细内容请看 4.11 节。当一个激活函数 $\varphi(v)$ 满足条件

$$\varphi(-v) = -\varphi(v)$$

我们说它是反对称的(即为它的自变量的奇函数)，见图 4-10a。在图 4-10b 的标准 logistic 函数不满足该条件。

关于反对称函数的一个非常流行的例子是一个双曲正切的 sigmoid 型非线性性，即

$$\varphi(v) = a\tanh(bv)$$

其中 a，b 是常数。合适的 a，b 值是(LeCun,1989,1993)

$$a = 1.7159$$

179

$$b = \frac{2}{3}$$

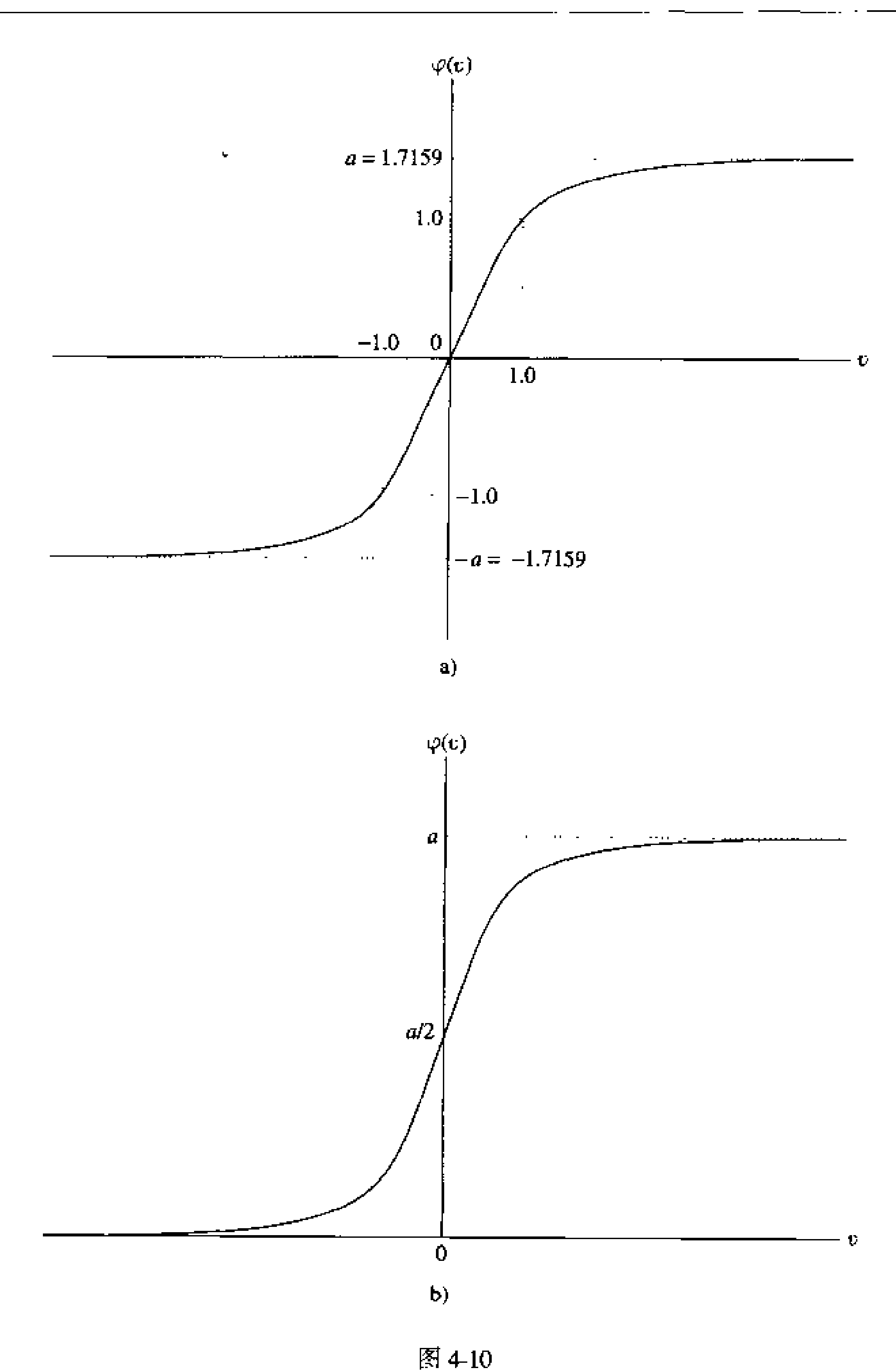


图 4-10
a)反对称激活函数 b)非对称激活函数

这样定义的双曲正切函数有如下有用的性质：

- $\varphi(1)=1$ 和 $\varphi(-1)=-1$。
- 在原点激活函数的倾斜度(即有效增益)接近于1，如下所示：

$$\varphi(0) = ab = 1.7159 \times 2/3 = 1.1424$$

- $\varphi(v)$的二阶导数在 $v=1$ 时达到最大。

4. 目标值。在 sigmoid 激活函数的范围内选择目标值(期望响应)是很重要的。特别地，多层感知器的输出层的神经元 j 的期望响应 d_j 必须被与 sigmoid 激活函数的极限值偏离某个 ε

值。否则反向传播算法会使网络的参数趋向于无穷大，驱使隐藏神经元达到饱和从而减慢学习过程。具体地，考虑图 4-10a 所示的反对称激活函数。对于极限值 $+a$，我们令

$$d_j = a - \varepsilon$$

对于有限值 $-a$，我们令

$$d_j = -a + \varepsilon$$

这里 ε 是一个合适的正常数。对前面选择的 $a = 1.7159$，可以令 $\varepsilon = 0.7159$，这样，目标值可以方便地选为 ± 1，见图 4-10a。

5. 输入规整化。每一个不同的输入变量都需要预处理，使得它关于整个训练集求平均的均值接近 0，或者与标准偏差相比是比较小的(LeCun, 1993)。为评价这个规则的实际意义，我们考虑当输入恒正时的极端情况。在这种情况下，第一隐藏层的一个神经元的所有突触权值只能同时增加或同时减少。所以，如果这个神经元权值向量改变方向，则它的误差曲面的路径变成锯齿形的，这会使收敛速率变慢，因此应该避免。

要加速反向传播学习的过程，输入变量的规整化必须包括下面两个步骤：

- 训练集包含的输入变量应该不相关的；这可以通过第 8 章提到的主分量分析法来做到。
- 去相关后的输入变量应调整其长度使得它们的协方差近似相等，因此可以保证网络中的不同突触权值以大约相等的速度进行学习。 181

图 4-11 说明依次执行规整化三个步骤的结果：消除均值，去相关性，以及协方差均衡。

6. 初始化。网络的突触权值和阈值初值的一个较好的选择对一个成功的网络设计会有巨大的帮助。关键问题是：什么是好的选择？

当突触权值被赋予一个较大的初始值，那么网络的神经元很可能会趋于饱和。如果发生这种情况，反向传播算法中的局域梯度呈现出一个很小的值，结果导致反向传播学习过程很缓慢。然而，如果突触权值被赋予一个较小的初始值，反向传播算法可能就在误差曲面的原点的一个非常平缓的区域内进行；特别对于反对称函数(如双曲正切函数)的条件下，这种可能性就更大。不幸地是，这个原点是一个鞍点，这个鞍点是一个稳定点，在该点处与马鞍正交的误差曲面的曲率为正，而沿着马鞍方向为负。由于这些原因，使用过大或过小值初始化突触权值都应该避免。恰当的初始化选择位于这两种极端之间。 182

具体地，考虑将一个双曲正切函数作为激活函数的多层感知器。设网络的每一个神经元偏置为 0。我们将神经元 j 的诱导局部域表示为

$$v_j = \sum_{i=1}^{m} w_{ji} y_i$$

假设网络的每一个神经元的输入的均值为 0 方差为 1，表示为

$$\mu_y = E[y_i] = 0 \qquad \text{对所有神经元 } i$$

和

$$\sigma_y^2 = E[(y_i - \mu_i)^2] = E[y_i^2] = 1 \qquad \text{对所有神经元 } i$$

进一步，假设输入值都是不相关的，即

$$E[y_i y_k] = \begin{cases} 1, & k = i \\ 0, & k \neq i \end{cases}$$

并且设突触权值的值是以均值为 0 的均匀分布抽取的一组数

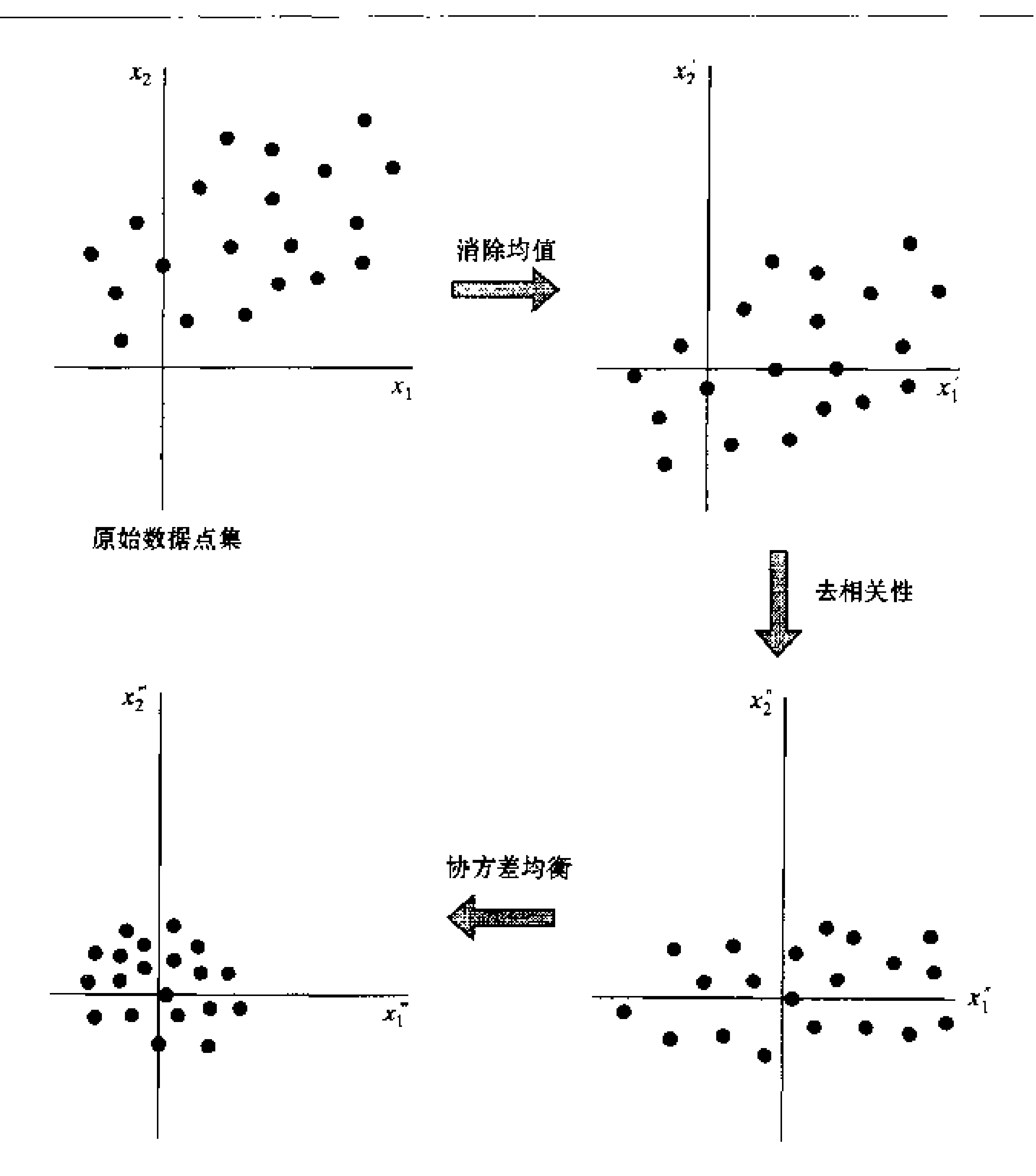


图 4-11 二维输入空间的消除均值、去相关性以及协方差均衡运算的图示

$$\mu_w = E[w_{ji}] = 0 \qquad \text{对所有}(j,i)\text{对}$$

和方差

$$\sigma_w^2 = E[(w_{ji} - \mu_w)^2] = E[w_{ji}^2] \qquad \text{对所有}(j,i)\text{对}$$

因此我们可以将诱导局部域 v_j 的均值和方差表示为

$$\mu_v = E[v_j] = E\Big[\sum_{i=1}^{m} w_{ji}y_i\Big] = \sum_{i=1}^{m} E[w_{ji}]E[y_i] = 0$$

和

$$\begin{aligned}\sigma_v^2 &= E[(v_j - \mu_v)^2] = E[v_j^2] = E\Big[\sum_{i=1}^{m}\sum_{k=1}^{m} w_{ji}w_{jk}y_iy_k\Big] \\ &= \sum_{i=1}^{m}\sum_{k=1}^{m} E[w_{ji}w_{jk}]E[y_iy_k] = \sum_{i=1}^{m} E[w_{ji}^2] = m\sigma_w^2\end{aligned} \tag{4.48}$$

183

这里 m 是一个神经元的突触连接的数目。

根据上述结果，我们对如何将突触权值初始化描述一个好策略，使得神经元诱导局部域的标准偏差位于它的 sigmoid 激活函数的线性部分和饱和部分的过渡区域。例如，如前所述的参数 a 和 b 所设值的双曲正切函数，当式(4.48)中的 $\sigma_v = 1$ 时可以满足这个目标，这样我

们得到

$$\sigma_u = m^{-1/2} \tag{4.49}$$

因此，对于一个均匀分布，它需要其均值为 0 而方差将与神经元的突触连接的数目成反比，从而以这个分布来选择突触权值的值。

7. 从提示中学习。从一组未知的训练例子中学习意味着处理未知的输入 - 输出映射函数 $f(\cdot)$。事实上，学习过程利用函数 $f(\cdot)$ 例子所包含的信息来推断它的逼近实现。从例子中学习的过程可以推广为包括从提示中学习，这可以由在学习过程中允许包括我们已有的关于函数 $f(\cdot)$ 的先验知识来实现(Abu-Mostafa, 1995)。这些知识包括不变性、对称性或关于函数 $f(\cdot)$ 的其他知识，它们可以用来加速 $f(\cdot)$ 的逼近实现的搜索，而且更重要的是，会提高最后估计的质量。式(4.49)的使用就是怎样取得这一点的例子。

8. 学习率。多层感知器的所有神经元理论上应以同一速率进行学习。网络的最后一层的局域梯度通常比别的层大。因此，最后一层的学习率参数 η 应设得比别的层小。有很多输入的神经元的学习率参数应比输入较少的神经元小。在 LeCun(1993)中提到对一个给定的神经元，其学习率应与该神经元的突触连接的平方根成反比。关于学习率我们将在 4.17 节中作更多的讨论。

4.7 输出表示和决策规则

理论上，一个 M 类分类问题中对 M 个不同类的并组成整个输入空间，我们需要 M 个输出表示所有可能的分类决策，如图 4-12 所描绘。在这个图中，向量 $\mathbf{x}_j$ 指由多层感知器分类的 m 维随机向量 $\mathbf{x}$ 的第 j 个原型(即，惟一的样本)。$\mathbf{x}$ 可以属于的 M 个可能类的第 k 类表示为 $\mathscr{C}_k$。用 y_{kj} 表示响应于 $\mathbf{x}_j$ 的网络的第 k 个输出神经元的输出，表示如下

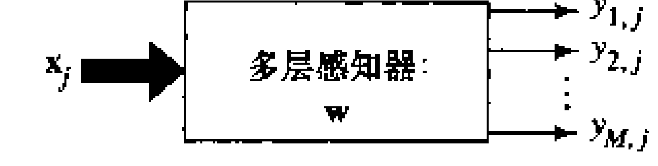


图 4-12 模式分类的方框图

$$y_{k,j} = F_k(\mathbf{x}_j), \qquad k = 1,2,\cdots,M \tag{4.50}$$

184

这里函数 $F_k(\cdot)$ 定义网络从输入到第 k 个输出所学习的映射。为表示方便起见，令

$$\mathbf{y}_j = [y_{1,j}, y_{2,j}, \cdots, y_{M,j}]^T = [F_1(\mathbf{x}_j), F_2(\mathbf{x}_j), \cdots, F_M(\mathbf{x}_j)]^T = \mathbf{F}(\mathbf{x}_j) \tag{4.51}$$

这里 $\mathbf{F}(\cdot)$ 是一个向量值函数。在这一节我们想解决的一个基本问题是：

在一个多层感知器被训练后，用于分类网络 M 个输出的最优决策规则应该是什么？

很清楚，任何合理的决策规则都应该建立在下述向量值函数的基础上：

$$\mathbf{F}: \mathbb{R}^m \ni \mathbf{x} \to \mathbf{y} \in \mathbb{R}^M \tag{4.52}$$

一般来说，关于向量值函数确定的一点是它是一个连续函数并使经验风险泛函最小：

$$R = \frac{1}{2N} \sum_{j=1}^{N} \| \mathbf{d}_j - \mathbf{F}(\mathbf{x}_j) \|^2 \tag{4.53}$$

这里 $\mathbf{d}_j$ 是原型 $\mathbf{x}_j$ 的期望(目标)输出模式，$\|\cdot\|$ 是所含向量的欧几里德范数，N 是输入网络进行训练的样本数目。式(4.53)准则的本质与式(4.3)的代价函数一致。向量值函数 $\mathbf{F}(\cdot)$ 强烈依赖于用于网络训练的例子 $(\mathbf{x}_j, \mathbf{d}_j)$，因此不同的 $(\mathbf{x}_j, \mathbf{d}_j)$ 值会导致不同的向量值函数 $\mathbf{F}(\cdot)$。注意，这里用到的 $(\mathbf{x}_j, \mathbf{d}_j)$ 术语和前面用到的 $(\mathbf{x}(j), \mathbf{d}(j))$ 相同。

假设现在用二值目标值来训练网络(当网络使用 logistic 函数时它恰巧对应于网络输出的

上限和下限)，可以写为

$$d_{kj}=\begin{cases}1 & \text{原型 } \mathbf{x}_j \text{ 属于类} \mathscr{C}_k \\ 0 & \text{原型 } \mathbf{x}_j \text{ 不属于类} \mathscr{C}_k\end{cases} \tag{4.54}$$

基于上面的解释，$\mathscr{C}_k$ 表示为 M 维目标向量

$$\begin{bmatrix}0\\ \vdots\\ 1\\ \vdots\\ 0\end{bmatrix} \leftarrow \text{第 } k \text{ 个元素}$$

对一组有限的相互独立且同分布(i.i.d)的训练样本使用反向传播算法训练多层感知分类器，185 一种富有吸引力的假设是该多层感知分类器可能得到固有的后验类概率的一个渐进近似。这个性质可由下面的理由证实(White,1989a;Richard and Lippmann,1991)：

- 利用大数定律证明当训练集的大小 N 趋于无穷大时，最小化式(4.53)中代价泛函 R 的权值向量 $\mathbf{w}$ 趋于使随机量$\frac{1}{2}\|\mathbf{d}-\mathbf{F}(\mathbf{w},\mathbf{x})\|^2$ 的期望最小的最优权值向量 $\mathbf{w}^*$，其中 $\mathbf{d}$ 是期望响应向量，$\mathbf{F}(\mathbf{w},\mathbf{x})$是输入为 $\mathbf{x}$ 时具有权值向量 $\mathbf{w}$ 的多层感知器所实现的逼近值(White,1989a)。函数 $\mathbf{F}(\mathbf{w},\mathbf{x})$明确表示对权值向量 $\mathbf{w}$ 的依赖，就是前面说的 $\mathbf{F}(\mathbf{x})$。
- 最优权值向量 $\mathbf{w}^*$ 使得网络实际输出 $\mathbf{F}(\mathbf{w}^*,\mathbf{x})$，是给定输入向量 $\mathbf{x}$ 期望响应向量的条件期望的均方误差最小的估计值(White,1989a)。这在第2章已经讨论过了。
- 对于1对 M 的模式分类问题，如果输入向量 $\mathbf{x}$ 属于$\mathscr{C}_k$，则期望响应向量的第 k 个元素等于1，其他分量为0。因此对于给定的 $\mathbf{x}$，期望响应向量的条件期望等于后验类概率 $P(\mathscr{C}_k|\mathbf{x})$，$k=1,2,\cdots,M$(Richard and Lippmann,1991)。

因此随之而来的是如果训练集足够大且反向传播算法没有陷入局域极小，则一个多层感知分类器(使用 logistic 函数非线性性)确实接近于后验类概率。我们现在可以回答前面提出的问题。具体地，我们可以说一个适当的输出决策规则是由后验概率估计产生的(近似)Bayes 规则：

$$\text{如果} \qquad F_k(\mathbf{x})>F_j(\mathbf{x}),\ \text{对所有 } j\neq k \tag{4.55}$$

将随机向量 $\mathbf{x}$ 分类为$\mathscr{C}_k$，这里 $F_k(\mathbf{x})$和 $F_j(\mathbf{x})$是下列向量值映射函数的分量：

$$\mathbf{F}(\mathbf{x})=\begin{bmatrix}F_1(\mathbf{x})\\ F_2(\mathbf{x})\\ \vdots\\ F_M(\mathbf{x})\end{bmatrix}$$

当固有的后验分类分布互不相同时，以概率1存在惟一的最大输出值。(这里假设使用无限精度计算；有限精度时才可能出现多于一个最大值的情形。)决策规则的优点是比基于输出"点火"概念选择类属关系的常用"特别"法则提供了一个更明确的决策。这里常用"特别"规则是指如果相应输出值比固定的阈值大(对 logistic 形的激活函数常用0.5)，向量 $\mathbf{x}$ 是赋值给特定的类属关系，这会导致多重类赋值。

在4.6节我们指出与式(4.30)的logistic函数相应的二值目标值[0,1]常用一个小的ε进行扰动后作为实际度量值，这样可以在网络的训练中避免突触权值的饱和(由于有限的数值精度)。作为这个扰动的结果，现在目标值是非二值的，而且渐进逼近$F_k(\mathbf{x})$不再精确是M类的一个后验概率$P(\mathscr{C}_k \mid \mathbf{x})$(Hampsire and Pearlmutter, 1990)。相反$P(\mathscr{C}_k \mid \mathbf{x})$线性映射到闭区间$[\varepsilon, 1-\varepsilon]$，使得$P(\mathscr{C}_k \mid \mathbf{x})=0$对应输出$\varepsilon$，而$P(\mathscr{C}_k \mid \mathbf{x})=1$对应$1-\varepsilon$。由于这个线性映射保持相对的顺序，它并不影响应用式(4.55)的决策规则的结果。 [186]

同样有趣的是，当一个决策边界由一个多层感知器的输出经过一些固定阈值判断形成时，决策边界的所有形状和方向可以试探地(对一个隐藏层的情形)用相应的隐藏神经元的数目和与之连接的突触权值的比来解释(Lui, 1990)。然而，这样的分析不能应用于根据式(4.55)的输出决策规则形成的决策边界。一个更合适的处理是将隐藏层神经元当成非线性特征检测器，它对原始输入空间$\mathbb{R}^{m_0}$(这里类之间可能并不是线性可分的)映射为在隐藏层激活输出的空间，此处它们更有可能是线性可分的。

4.8 计算机实验

在这一节我们用计算机实验来说明多层感知器作为模式分类器的学习行为。实验的目标是区别两类"重叠"的二维Gauss分布模式(标号为1和2)。用$\mathscr{C}_1$和$\mathscr{C}_2$分别表示随机向量$\mathbf{x}$属于模式1和2的事件集合。然后，我们可以分别表示这两类的条件概率密度函数：

类$\mathscr{C}_1$：
$$f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_1)=\frac{1}{2\pi\sigma_1^2}\exp\left(-\frac{1}{2\sigma_1^2}\|\mathbf{x}-\boldsymbol{\mu}_1\|^2\right) \tag{4.56}$$

其中，$\boldsymbol{\mu}_1$=均值向量$=[0,\ 0]^T$，σ_1^2=方差$=1$

类$\mathscr{C}_2$：
$$f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_2)=\frac{1}{2\pi\sigma_2^2}\exp\left(-\frac{1}{2\sigma_2^2}\|\mathbf{x}-\boldsymbol{\mu}_2\|^2\right) \tag{4.57}$$

其中，$\boldsymbol{\mu}_2=[2,\ 0]^T$，$\sigma_2^2=4$

假设这两类是等概率的，即

$$p_1 = p_2 = \frac{1}{2}$$

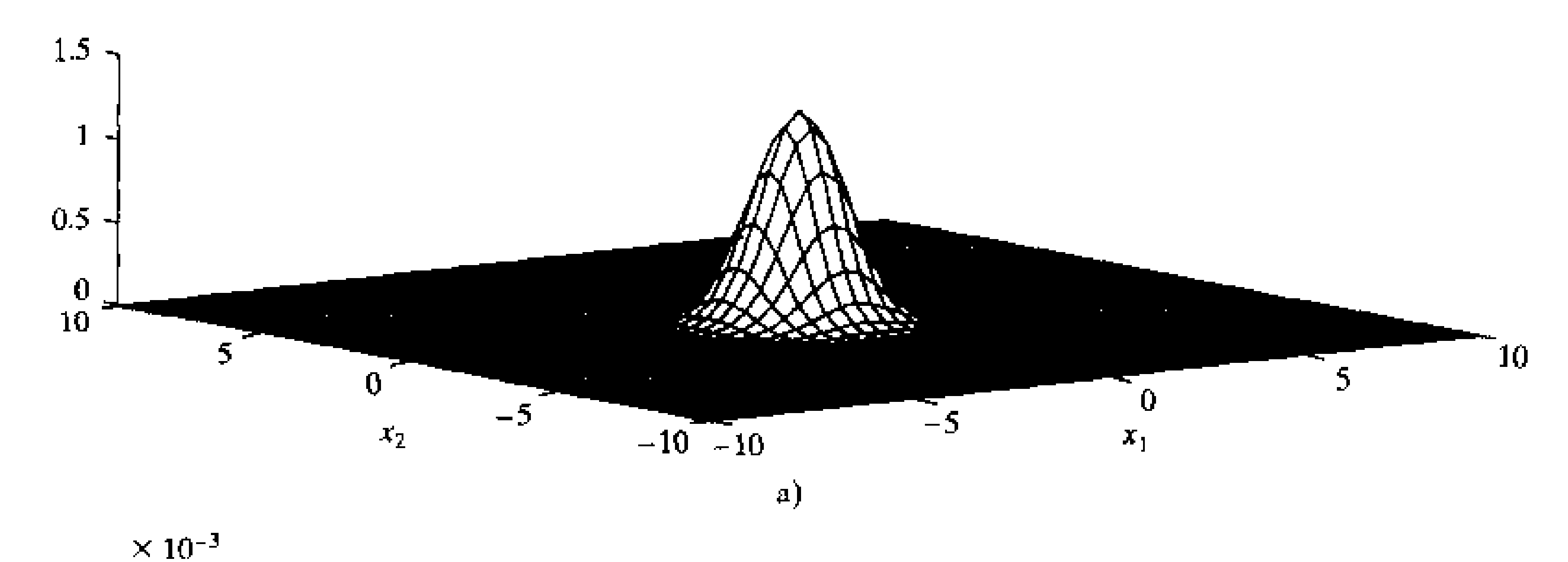
图4-13a分别表示了式(4.56)和(4.57)两类Gauss分布的三维图。输入向量是$\mathbf{x}=[x_1, x_2]^T$，且输入空间的维数是$m_0=2$。图4-14是类1和类2的各自的散布图和它们的联合散布图，图中分别从两个过程中选取了500个点。后一个图清楚地表示两种分布的重叠，这表明无可避免会有明显的分类错误概率。 [187]

Bayesian决策边界

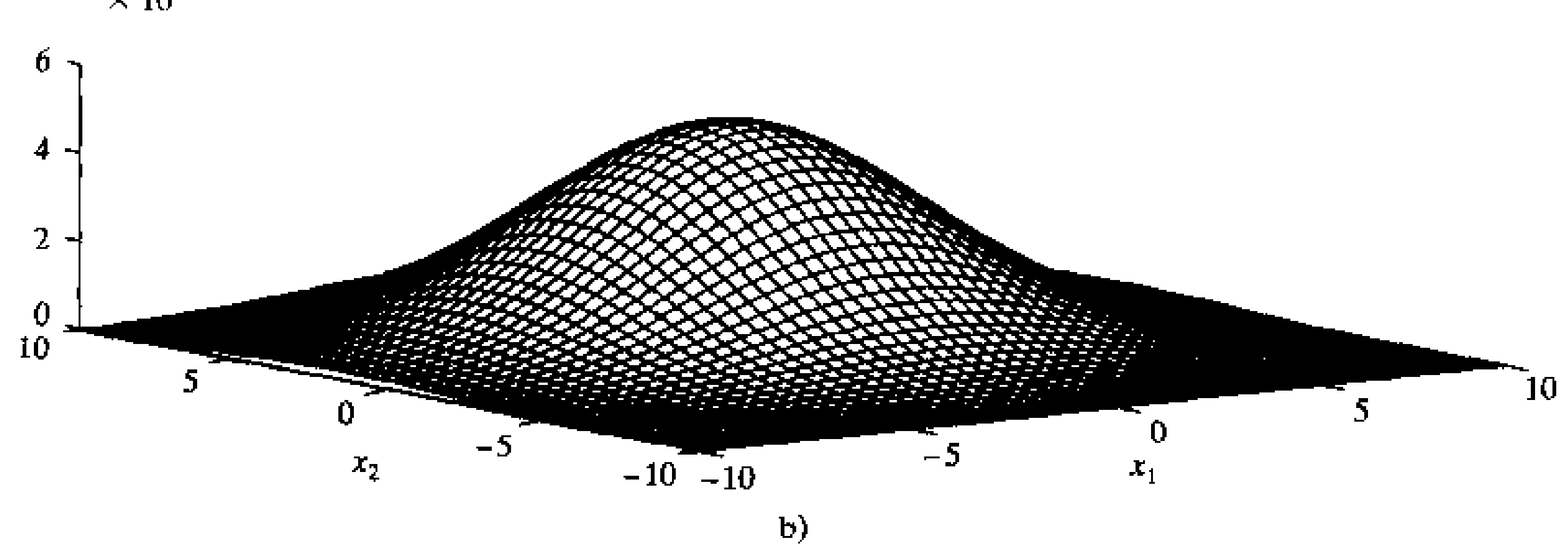
最优分类的Bayes准则在第3章中已经讨论过了。假设那是一个两类问题，(1)类$\mathscr{C}_1$和类$\mathscr{C}_2$等概率，(2)正确分类的代价为0，(3)错误分类的代价是相等的，我们发现最优决策边界是利用似然比检验：

$$\Lambda(\mathbf{x}) \underset{\mathscr{C}_1}{\overset{\mathscr{C}_2}{\lessgtr}} \xi \tag{4.58}$$

这里$\Lambda(\mathbf{x})$是似然比，定义为



a)



b)

图 4-13
a)概率密度函数 $f_{\mathbf{x}}(\mathbf{x}|\mathscr{C}_1)$ b)概率密度函数 $f_{\mathbf{x}}(\mathbf{x}|\mathscr{C}_2)$

188

$$\Lambda(\mathbf{x}) = \frac{f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_1)}{f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_2)} \tag{4.59}$$

ξ是检验的阈值，定义为

$$\xi = \frac{p_2}{p_1} = 1 \tag{4.60}$$

对考虑的例子，我们有

$$\Lambda(\mathbf{x}) = \frac{\sigma_2^2}{\sigma_1^2}\exp\left(-\frac{1}{2\sigma_1^2}\|\mathbf{x}-\boldsymbol{\mu}_1\|^2 + \frac{1}{2\sigma_2^2}\|\mathbf{x}-\boldsymbol{\mu}_2\|^2\right)$$

因此，最优(Bayes)决策边界由

$$\frac{\sigma_2^2}{\sigma_1^2}\exp\left(-\frac{1}{2\sigma_1^2}\|\mathbf{x}-\boldsymbol{\mu}_1\|^2 + \frac{1}{2\sigma_2^2}\|\mathbf{x}-\boldsymbol{\mu}_2\|^2\right) = 1$$

定义，或者等价地定义为

$$\frac{1}{\sigma_2^2}\|\mathbf{x}-\boldsymbol{\mu}_2\|^2 - \frac{1}{\sigma_1^2}\|\mathbf{x}-\boldsymbol{\mu}_1\|^2 = 4\log\left(\frac{\sigma_1}{\sigma_2}\right) \tag{4.61}$$

利用简单的运算，可以将式(4.61)简化为

$$\|\mathbf{x}-\mathbf{x}_c\|^2 = r^2 \tag{4.62}$$

这里

$$\mathbf{x}_c = \frac{\sigma_2^2\boldsymbol{\mu}_1 - \sigma_1^2\boldsymbol{\mu}_2}{\sigma_2^2 - \sigma_1^2} \tag{4.63}$$

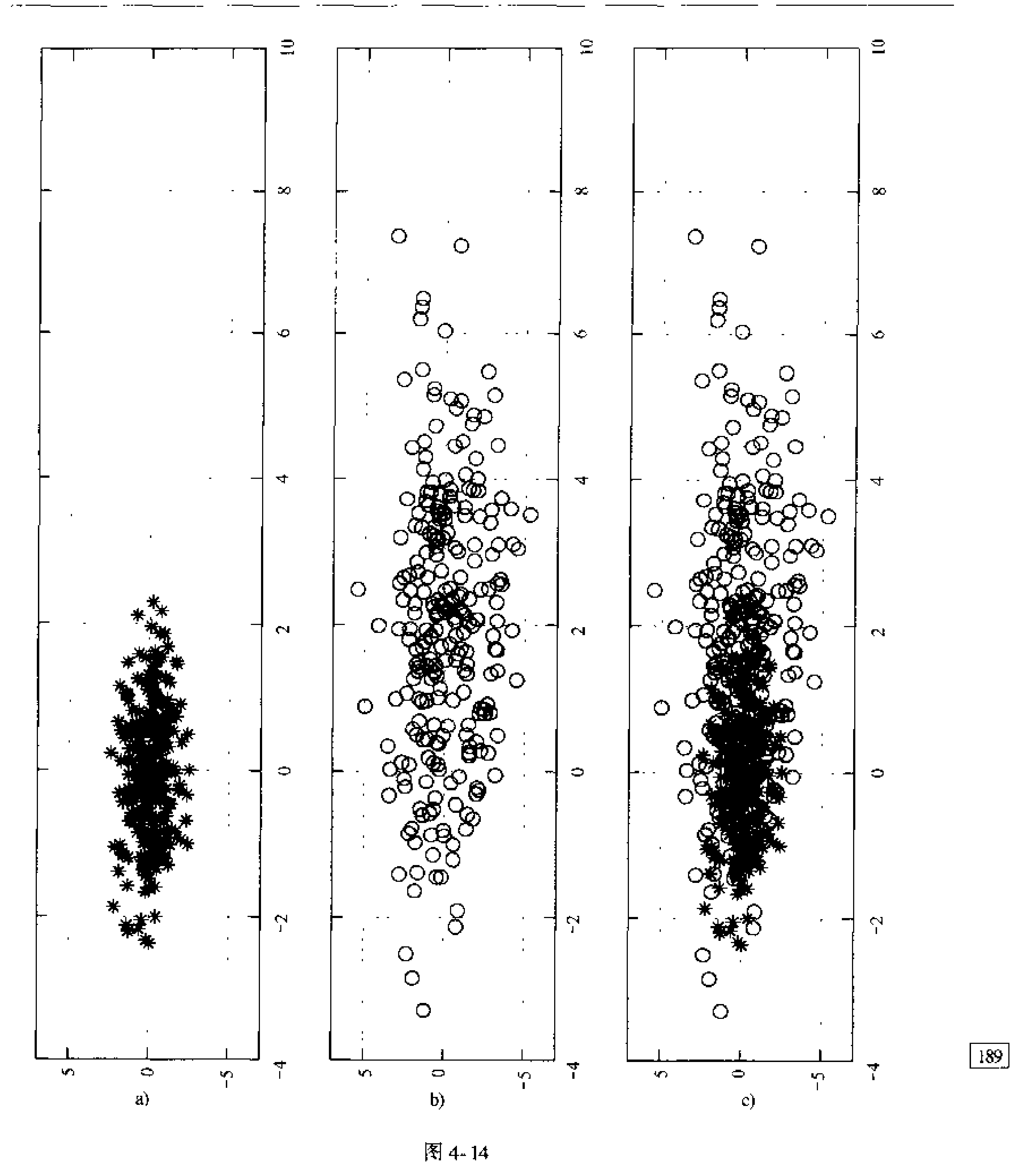


图 4-14

a)类$\mathscr{C}_1$ 的散布图　b)类$\mathscr{C}_2$ 的散布图　c)类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 的总体散布图

189

和

$$r^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_2^2 - \sigma_1^2}\left[\frac{\| \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 \|^2}{\sigma_2^2 - \sigma_1^2} + 4\log\left(\frac{\sigma_2}{\sigma_1}\right)\right] \tag{4.64}$$

式(4.62)代表以 $\mathbf{x}_c$ 为圆心和 r 为半径的一个圆。令 Ω_1 定义为这个圆内的区域。对当前问题 Bayes 分类规则可陈述如下：

如果似然比 $\Lambda(\mathbf{x})$ 比阈值 ξ 大，则将观察向量 $\mathbf{x}$ 分类到类$\mathscr{C}_1$，否则就分类到类$\mathscr{C}_2$。

190

对于这个实验的特殊参数，我们有圆形决策边界，其圆心位于 $\mathbf{x}_c = \begin{bmatrix} -2/3 \\ 0 \end{bmatrix}$，其半径为 $r \simeq 2.34$。

用 c 来表示正确分类结果的集合，e 表示错误分类结果的集合。根据 Bayes 决策规则运行的分类器错误(错误分类)概率 P_e 是

$$P_e = p_1 P(e \mid \mathscr{C}_1) + p_2 P(e \mid \mathscr{C}_2) \tag{4.65}$$

这里 $P(e|\mathscr{C}_1)$是给定分类输入向量来自于类$\mathscr{C}_1$ 时的错误分类的条件概率，$P(e|\mathscr{C}_2)$类似；p_1 和 p_2 分别为类$\mathscr{C}_1$ 和$\mathscr{C}_2$ 的先验概率。对于我们的问题，可以从数值上估计概率积分，得到

$$P(e \mid \mathscr{C}_1) \simeq 0.1056$$

$$P(e \mid \mathscr{C}_2) \simeq 0.2642$$

又有 $p_1 = p_2 = 1/2$，所以错误分类的概率是

$$P_e \simeq 0.1849$$

等价地，正确分类的概率为

$$P_c = 1 - P_e \simeq 0.8151$$

最优多层感知器的实验确定

表 4-1 列出多层感知器的各种可变参数，包括一个单层隐藏神经元，它是用反向传播算法以串行方式训练的。因为模式分类的最终目标是达到可接受的正确分类率，这个准则用于判断何时 MLP(用作一个模式分类器)的各种可变参数是最优的。

表 4-1 多层感知器的可变参数

191

参 数	符 号	典型变化范围
隐藏神经元数目	m_1	$(2, \infty)$
学习率参数	η	(0, 1)
动量常数	α	(0, 1)

隐藏神经元的最优数目 在实际处理时对于决定隐藏神经元的最优数目 m_1 的问题，利用的准则是能够产生与 Bayes 分类器性能“接近”(通常差 1%)的隐藏层神经元的最小数目作为最优隐藏神经元数目。因此，实验研究开始于两个隐藏层神经元作为起始点，模拟结果列在表 4-2 中。因为第一组模拟的功能是仅仅确定两个隐藏层神经元是否足够，学习率参数 η 和动量常数 α 被赋予任意平常的值。在每一个模拟过程进行时，对类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 以相同的概率随机产生 Gauss 分布训练例子，它们通过网络重复循环，每一个网络循环代表一个回合。回合的数目的选择是要使每次运行的训练例子总数为一个常数。这样做，由于训练集大小的变化而产生的潜在影响就平均掉了。

表 4-2 两个隐藏神经元的模拟结果*

运行号	训练集数目	回合数目	均方误差	正确分类概率 P_c
1	500	320	0.2375	80.36%
2	2000	80	0.2341	80.33%
3	8000	20	0.2244	80.47%

* 学习率 $\eta = 0.1$ 和动量 $\alpha = 0$。

在表4-2和下面的表中，均方误差是由式(4.53)定义的函数精确计算的。我们强调在这些表中包括均方误差仅仅把它当作一个记录，因为一个小的均方误差并非必然隐含好的泛化能力(即对从来没有遇到的数据有好的性能)。

在用 N 个模式训练网络收敛以后，正确分类的概率理论上可以计算如下：

$$P(c,N) = p_1 P(c,N \mid \mathscr{C}_1) + p_2 P(c,N \mid \mathscr{C}_2) \quad (4.66)$$

这里 $p_1 = p_2 = 1/2$，且

$$P(c,N \mid \mathscr{C}_1) = \int_{\Omega_1(N)} f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_1)\, d\mathbf{x} \quad (4.67)$$

$$P(c,N \mid \mathscr{C}_2) = 1 - \int_{\Omega_1(N)} f_{\mathbf{x}}(\mathbf{x} \mid \mathscr{C}_2)\, d\mathbf{x} \quad (4.68)$$

而 $\Omega_1(N)$ 是决策域空间区域，对这个区域的向量 $\mathbf{x}$(代表随机向量 $\mathbf{X}$ 的一次实现)多层感知器(用 N 个模式训练后)将它分到类 $\mathscr{C}_1$。这个区域通常由试验发现，计算网络学会的映射函数值，然后运用式(4.55)的输出决策规则就可以找出这个区域。不幸的是，$P(c,N|\mathscr{C}_1)$ 和 $P(c,N|\mathscr{C}_2)$ 的数值估计是一个问题，因为描述决策域 $\Omega_1(N)$ 的封闭形式的表达式并不容易找到。

因此，我们转而求助于实验逼近，涉及对训练后的多层感知器检验另外的独立例子集，这些例子是也是独立地以相同概率从类 $\mathscr{C}_1$ 和类 $\mathscr{C}_2$ 的分布中随机抽取的。令 A 为随机变量表示从 N 个实验模式中正确分类的模式数。因此比率 [192]

$$p_N = \frac{A}{N}$$

是一个随机变量，它提供了网络实际分类性能 p 的最大似然无偏估计。假设关于 N 对输入－输出而言 p 是一个常数，我们可以将 Chernoff 界(Devroye,1991)用于 p 的估计 p_N，得到

$$P(|p_N - p| > \varepsilon) < 2\exp(-2\varepsilon^2 N) = \delta$$

对于 $\varepsilon = 0.01$，$\delta = 0.01$(即以99%的概率保证对 p 的估计具有给定的容忍度)应用 Chernoff 界得到 $N \simeq 26\,500$。因此，我们挑选一个 $N = 32\,000$ 的测试集。表4-2的最后一列给出这个测试集的正确分类概率的估计，每一个结果都为试验的十个独立实现的平均值。

在表4-2中列出的有两个隐藏层的多层感知器的分类性能已经合理地接近于 Bayes 性能 $P_c = 81.51\%$。在这种基础上，我们可以总结出对于这里描述的模式分类问题使用两个隐藏神经元是合适的。为了强调这个结论，在表4-3中列出有四个隐藏神经元的感知器的模拟结果，网络其他一些参数的值保持不变。虽然在表4-3中对于4个隐藏神经元均方误差比表4-2中对2个神经元的略小，但是正确分类的平均率并没有改进；事实上，还略微差了一点。对于这里描述的计算机实验的以后部分，隐藏层的数目保持为2。

表4-3 使用四个隐藏神经元的多层感知器的模拟结果*

运行号	训练集数目	回合数目	均方误差	正确分类概率 P_c
1	500	320	0.2199	80.80%
2	2000	80	0.2108	80.81%
3	8000	20	0.2142	80.19%

* 学习率 $\eta = 0.1$ 和动量 $\alpha = 0$。

[193]

最优学习和动量常数 对于学习率参数 η 和动量常数 α 的最优值，我们可以使用下面三

个定义中的任何一个：

1. 最优的 η 和 α 平均上使得网络收敛于误差曲面上局部最小所需回合数目最小。

2. 最优的 η 和 α 平均上或最坏情况下使得网络收敛于误差曲面上全局最小所需回合数目最小。

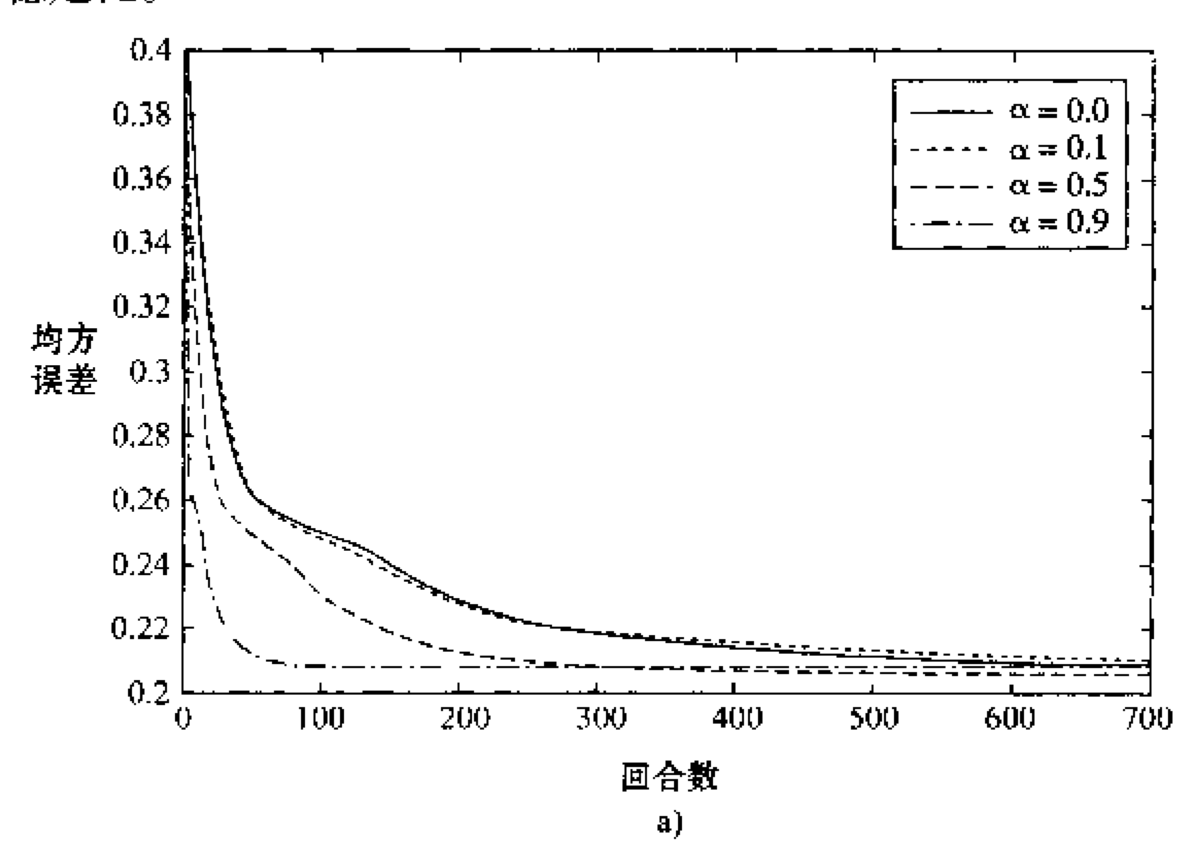
3. 最优的 η 和 α 平均上以最少的回合数使得网络收敛于在整个输入空间具有最好的泛化性能的网络配置。

这里使用的术语"平均"和"最坏情况"指的是训练输入－输出对的分布。定义3实际上是理想情况；然而很难应用因为在网络训练过程中最小化均方差通常是最优化的数学准则，而且正如前面所说，在一个训练集上较小的均方差并不意味着更好的泛化能力。从研究的观点来看，定义2比定义1更有意义。比如在Luo(1991)中给出关于学习率 η 的最佳适应值的严格结果，学习率 η 的最佳适应值指使得多层感知器估计全局最优突触权值矩阵达到期望的精度所使用的回合数最少的学习率 η 的值，虽然只是对线性神经元这种特殊情况。然而通常在使用定义1时，试探方法和实验性的过程决定了 η 和 α 的最优选择。因此对于这里描述的实验，在某种意义上我们认为是在定义1的意义下最优。

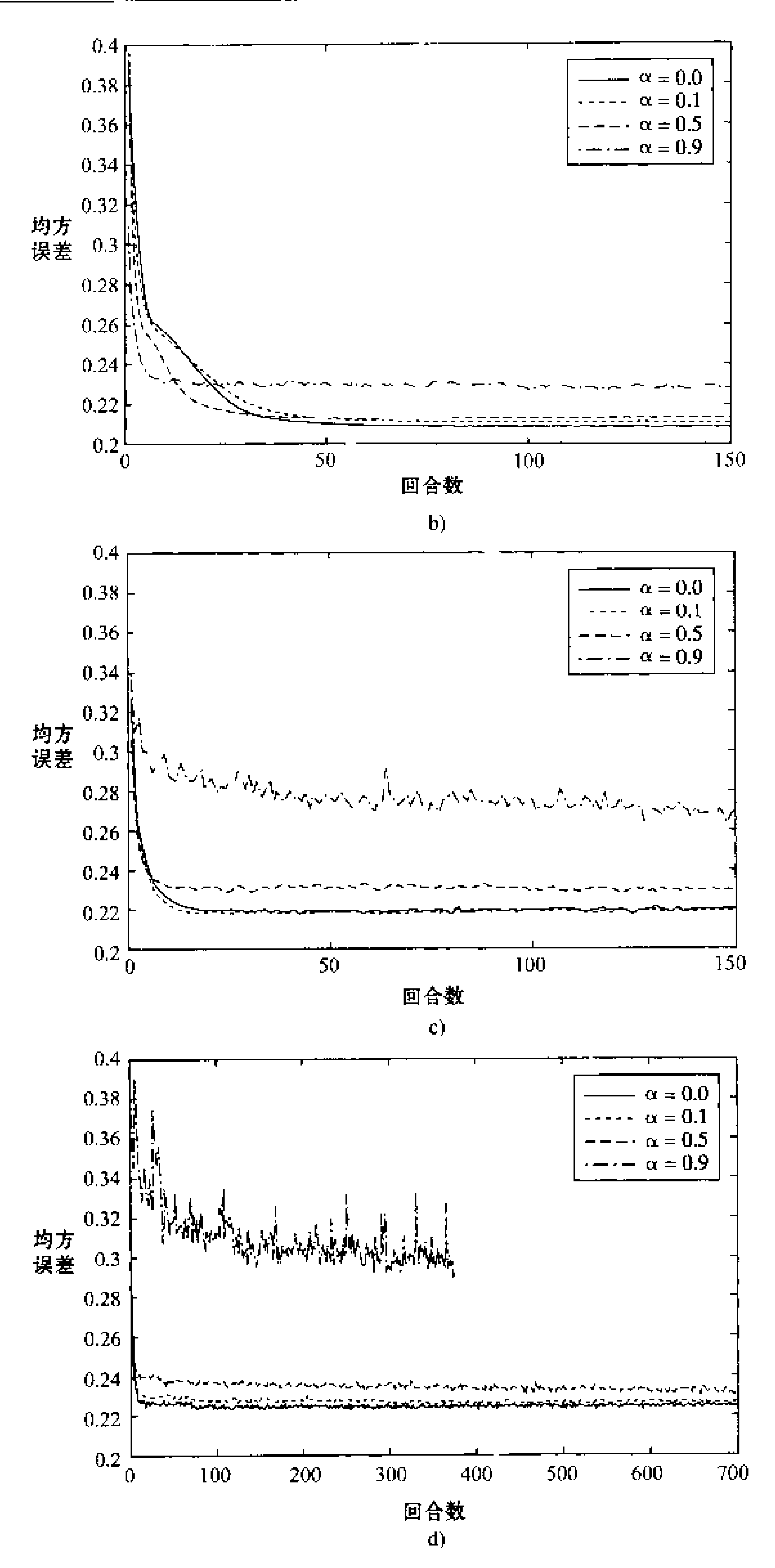
使用一个多层感知器和两个隐藏神经元，对学习率参数 $\eta\in\{0.01,0.1,0.5,0.9\}$ 和动量常数 $\alpha\in\{0.0,0.1,0.5,0.9\}$ 的组合进行模拟以观察它们在网络收敛上的效果。每个组合用相同的初始随机权值集和相同的500个样本集来训练，以便实验结果可以直接比较。学习过程连续进行700回合后结束；这个训练长度对于反向传播算法来说被认为是在误差曲面上足以达到局部最小值。这样计算的总体－平均学习曲线如图4-15a－4-15d所示，这些图是以 η 来单独分组的。

这里显示的实验性学习曲线指出如下的趋势：

- 通常当一个小的学习率参数 η 产生一个较慢的收敛时，它可以比一个大的 η 找到"更深"的局部最小值(在误差曲面中)。这个结果在直观上是令人满意的，因为一个小的 η 意味着一个最小值的搜索将会比在大的 η 的情况下覆盖更多的误差曲面。
- 194 当 $\eta\to0$ 时，使用 $\alpha\to1$ 使收敛速率加快。另一方面，当 $\eta\to1$ 时，又要求 $\alpha\to0$ 来保证学习的稳定性。



a)



195

图 4-15 对不同的动量 α 和学习率参数的下列值的总体平均学习曲线：a) $\eta=0.01$ b) $\eta=0.1$ c) $\eta=0.5$ d) $\eta=0.9$

- 常数 $\eta=\{0.5,0.9\}$ 和 $\alpha=0.9$ 的使用导致在学习过程中均方差的振荡以及在收敛时产生更大的均方差值，而这两种情况都不是期望的效果。

在图 4-16 中，我们显示"最佳"的学习曲线，这些学习曲线是从图 4-16 中各组学习曲线中选择出来的，以便决定一个整体上的最佳学习曲线，这里的"最佳"是从前面所描述的点 1 意义上定义的。图 4-16 显示最优学习率参数 η_{opt} 大约为 0.1，而最优动量常数 α_{opt} 大约为 0.5。因此，表 4-4 总结在其余实验中使用的网络参数最优值。图 4-16 中每条曲线的最终均方误差在 η 和 α 的范围上变化并不明显这一事实，暗示该问题有一个"表现良好"(即相当平滑)的误差曲面。

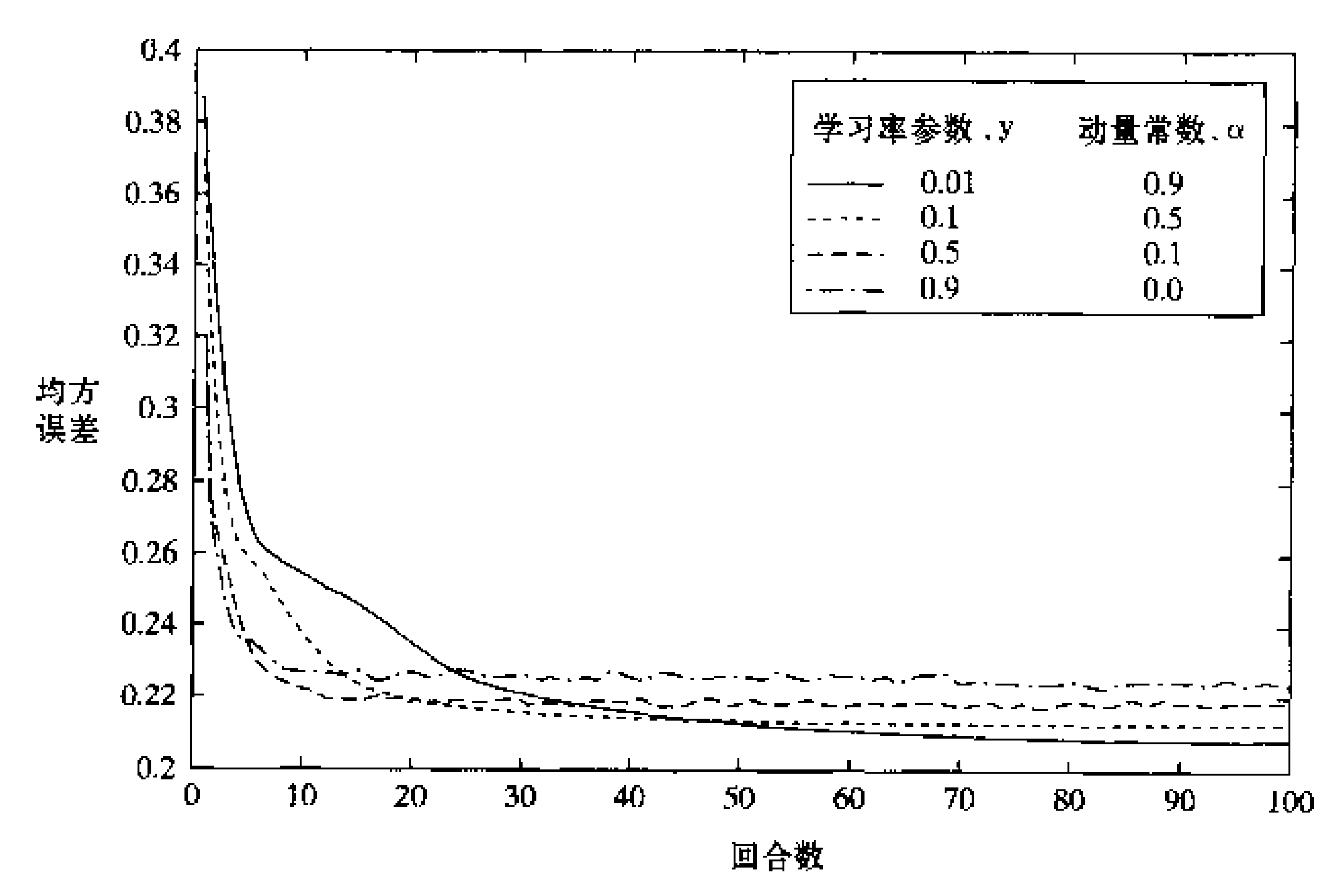


图 4-16　从图 4-15 的 4 部分挑选出的最好学习曲线

表 4-4　最优的多层感知器设置

参　数	符　号	值
神经元数目	m_{opt}	2
学习率参数	η_{opt}	0.1
动量常数	α_{opt}	0.5

最优网络设计的评价　给定的"最优"多层感知器具有如表 4-4 总结的参数，求出确定决策边界、总体-平均学习曲线以及正确分类的概率的最终网络的值。因为训练集有限，具有最优参数所学得的网络函数在本质上是"随机的"。因此这些性能度量是在 20 个独立训练网络之上的总体平均。每个训练集由 1000 个样本组成，这 1000 个样本是从 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 类的分布中以相同概率抽取出来的，并以随机顺序呈现给网络。和以前一样，训练持续 700 个回合，为了正确分类概率的实验性确定，先前曾使用过的 32 000 个例子的测试集再次被使用。

196

图 4-17a 显示在总体为 20 的 3 个网络的 3 个最佳决策边界；图 4-17b 显示在同样的总体中另外 3 个网络的 3 个最差决策边界。阴影(圆)的 Bayes 决策边界包含在两个图中以便参考。从这些图我们观察到由反向传播算法构建的决策边界相对于属于类 $\mathscr{C}_1$ 或 $\mathscr{C}_2$ 的区域而言是凸的，这里属于类 $\mathscr{C}_1$ 或 $\mathscr{C}_2$ 的区域是指决策边界将观察向量 $\mathbf{x}$ 归类到 $\mathscr{C}_1$ 或 $\mathscr{C}_2$ 类的区域。

197

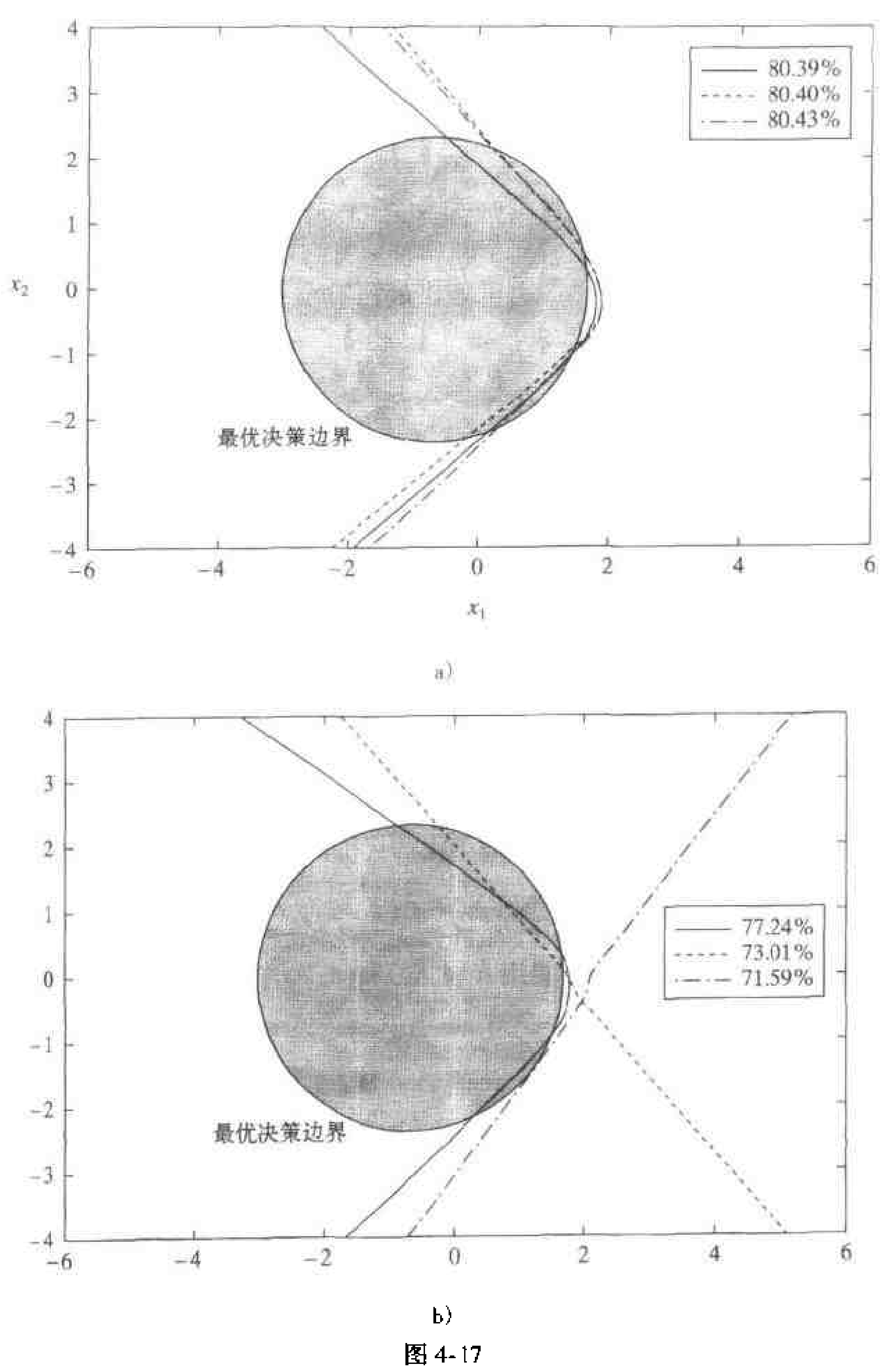


图 4-17
a)3 个分类精度最好的决策边界图：80.39%，80.40% 和 80.43%
b)3 个分类精度最差的决策边界图：77.24%，73.01% 和 71.59%

由训练样本计算出来的性能度量的总体统计特性、正确分类概率以及最终均方差罗列于表 4-5 中。对最佳 Bayes 分类器的正确分类概率为 81.51%。

表 4-5 性能度量的总体统计特性(样本数 = 20)

性能度量	均　值	标准偏差
正确分类概率	79.70%	0.44%
最终均方误差	0.2277	0.0118

198

4.9 特征检测

在采用反向传播算法学习的多层感知器的运算中，隐藏神经元具有重要的作用，这是因为隐藏神经元扮演着特征检测器的角色。随着学习过程的进行，隐藏神经元逐渐“发现”表征训练数据的潜在特征。它们之所以这样是通过执行一种非线性变换将输入数据变换到一种称之为隐藏空间或特征空间的新空间，隐藏空间或特征空间这两个术语在本书中互换使用。例如在模式分类任务下新空间中感兴趣的类可能比最初的输入空间更易彼此分离。4.5 节所讨论的 XOR 问题很好的说明了这一点。

为了把问题放置到数学环境下分析，假设一个多层感知器有一个包含 m_1 个隐藏神经元的非线性层，以及一个包含 $m_2 = M$ 个输出神经元的线性层。输出层中选择线性神经元的动机是希望集中注意力于隐藏神经元对多层感知器运行的作用。对网络突触权值进行调节，使网络的目标输出与实际输出之间的均方误差达到最小化，这里的目标输出是期望响应，实际输出是指为了响应 m_0 维输入向量(模式)，用对总共 N 个模式执行总体平均产生的输出。令 $z_j(n)$为隐藏神经元 j 在输入模式 n 下产生的输出。由于嵌入每个隐藏神经元的 sigmoid 激活函数，$z_j(n)$是应用于网络输入层的模式(向量)的一个非线性函数。

在输出层中神经元 k 的输出为

$$y_k(n) = \sum_{j=0}^{m_1} w_{kj} z_j(n), \qquad \begin{matrix} k = 1,2,\cdots,M \\ n = 1,2,\cdots,N \end{matrix} \tag{4.69}$$

这里 w_{k0}表示应用于神经元 k 的偏置。被最小化的代价函数为

$$\mathscr{E}_{av} = \frac{1}{2N} \sum_{n=1}^{N} \sum_{k=1}^{M} (d_k(n) - y_k(n))^2 \tag{4.70}$$

注意这里假定使用运行的集中方式。利用式(4.69)和(4.70)，容易对代价函数$\mathscr{E}_{av}$以紧凑矩阵形式重写为

$$\mathscr{E}_{av} = \frac{1}{2N} \| \tilde{\mathbf{D}} - \mathbf{W}\tilde{\mathbf{Z}} \|^2 \tag{4.71}$$

这里 $\mathbf{W}$ 是网络输出层突触权值的 $M \times m_1$ 矩阵。矩阵 $\tilde{\mathbf{Z}}$ 是隐藏神经元输出(减去了它们的平均值)的 $m_1 \times N$ 矩阵，它通过应用于网络输入层的 N 个输入模式生成，也即

$$\tilde{\mathbf{Z}} = \{(z_j(n) - \mu_{z_j}); j = 1,2,\cdots,m_1; n = 1,2,\cdots,N\}$$

这里 μ_{z_j} 是 $z_j(n)$的平均值。相应地，矩阵 $\tilde{\mathbf{D}}$ 是呈现给网络输出层的目标模式(期望响应)的 $M \times N$矩阵，也即

199

$$\tilde{\mathbf{D}} = \{(d_k(n) - \mu_{d_k}); k = 1,2,\cdots,M; n = 1,2,\cdots,N\}$$

这里 μ_{d_k} 是 $d_k(n)$的均值。认识到由式(4.70)定义的$\mathscr{E}_{av}$的最小化是一个线性最小平方问题，其解由

$$\mathbf{W} = \tilde{\mathbf{D}}\tilde{\mathbf{Z}}^+ \tag{4.72}$$

给出，这里 $\tilde{\mathbf{Z}}^+$ 是 $\tilde{\mathbf{Z}}$ 矩阵的伪逆矩阵。$\mathscr{E}_{av}$最小值如下(见习题 4.7)：

$$\mathscr{E}_{av,min} = \frac{1}{2N} \mathrm{tr}[\tilde{\mathbf{D}}\tilde{\mathbf{D}}^T - \tilde{\mathbf{D}}\tilde{\mathbf{Z}}^T(\tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^T) + \tilde{\mathbf{Z}}\tilde{\mathbf{D}}^T] \tag{4.73}$$

这里 tr[·]表示迹算子。因为用矩阵 $\tilde{\mathbf{D}}$ 表示的目标模式固定，根据多层感知器的突触权值来

最小化代价函数$\mathscr{E}_{av}$等价于最大化判别函数(Webb and Lowe,1990)

$$\mathscr{D} = \text{tr}[\mathbf{C}_b \mathbf{C}_t^+] \tag{4.74}$$

这里矩阵$\mathbf{C}_b$和$\mathbf{C}_t$定义如下:

- $m_1 \times m_1$矩阵$\mathbf{C}_t$是根据N输入模式得到的隐藏神经元输出的总体协方差矩阵

$$\mathbf{C}_t = \tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^T \tag{4.75}$$

矩阵$\mathbf{C}_t^+$是$\mathbf{C}_t$的伪逆矩阵。

- $m_1 \times m_1$矩阵$\mathbf{C}_b$定义为

$$\mathbf{C}_b = \tilde{\mathbf{Z}}\tilde{\mathbf{D}}^T\tilde{\mathbf{D}}\tilde{\mathbf{Z}}^T \tag{4.76}$$

注意由式(4.74)定义的判别函数$\mathscr{D}$完全由多层感知器的隐藏神经元决定。并且没有对组成非线性变换的隐藏层的层数有所限制，其中非线性变换负责生成判别函数$\mathscr{D}$。在隐藏层数目大于1的多层感知器中，矩阵$\tilde{\mathbf{Z}}$表示由最后隐藏神经元定义的空间中全部模式集。

为了对矩阵$\mathbf{C}_b$做出解释，考虑一个M选1(one-from-M)编码格式的特殊选择(Webb and Lowe,1990)。就是说，若所选模式属于那个类，则对该模式的目标值(期望响应)输出为1，否则为0，如下所示:

$$d(n) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow \text{第 } k \text{ 个元素}, \qquad d(n) \in \mathscr{C}_k$$

200

因此，假如有M个类$\mathscr{C}_k$，$k=1, 2, \cdots, M$，其中N_k个模式在类$\mathscr{C}_k$中并且有

$$\sum_{k=1}^{M} N_k = N$$

我们因而可以对这个特殊编码方案将矩阵$\mathbf{C}_b$展开为如下形式:

$$\mathbf{C}_b = \sum_{k=1}^{M} N_k^2 (\boldsymbol{\mu}_{z,k} - \boldsymbol{\mu}_z)(\boldsymbol{\mu}_{z,k} - \boldsymbol{\mu}_z)^T \tag{4.77}$$

这里$m_1 \times 1$的向量$\boldsymbol{\mu}_{z,k}$是隐藏神经元输出关于类$\mathscr{C}_k$中N_k个模式的向量平均值，而向量$\boldsymbol{\mu}_z$是隐藏神经元输出关于N个输入向量的向量平均值。根据式(4.77)，我们可以将$\mathbf{C}_b$解释为隐藏层输出的加权类间协方差矩阵。

因此，对于一个M选1的编码方案，多层感知器最大化一个判别函数，该判别函数为加权类间协方差矩阵和总体协方差矩阵的伪逆这两个矩阵乘积的迹。这个结果非常有趣，这是因为它说明一个由反向传播学习的多层感应器是如何融合单个类中的样本比例作为先验知识。

和 Fisher 线性判别式的关系

由式(4.74)定义的判别函数$\mathscr{D}$对于多层感知器来说是惟一的，它与 Fisher 的线性判别式非常相似，Fisher 的线性判别式描述一个由多维问题到一维问题的线性变换。假设变量y由

一个输入向量 $\mathbf{x}$ 的元素线性组合而成，也就是说 y 定义为 $\mathbf{x}$ 和可调参数 $\mathbf{w}$(包括一个偏置为其第一个元素)的向量的内积，所示如下：

$$y = \mathbf{w}^T\mathbf{x}$$

向量 $\mathbf{x}$ 是从类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 总体中的一个抽取出来的，类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 的总体由于它们的均值向量 $\boldsymbol{\mu}_1$ 和 $\boldsymbol{\mu}_2$ 不同而区别。区别这两个类的 Fisher 准则定义如下：

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{C}_b\mathbf{w}}{\mathbf{w}^T\mathbf{C}_t\mathbf{w}}$$

这里 $\mathbf{C}_b$ 是类间协方差矩阵，定义为

$$\mathbf{C}_b = (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^T$$

而 $\mathbf{C}_t$ 是总的类内协方差矩阵，定义为

201

$$\mathbf{C}_t = \sum_{n\in\mathscr{C}_1}(\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^T + \sum_{n\in\mathscr{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^T$$

类内协方差矩阵 $\mathbf{C}_t$ 与训练集的样本协方差矩阵成比列。它是对称的而且非负定，在训练集足够大时通常是非奇异矩阵。类间协方差矩阵 $\mathbf{C}_b$ 也是对称和非负定的，但它是奇异矩阵。一个特别有趣的性质是矩阵乘积 $\mathbf{C}_b\mathbf{w}$ 总是均值向量差 $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$ 的方向。这个特性由 $\mathbf{C}_b$ 定义直接得出。

我们知道定义 $J(\mathbf{w})$的表达式通称为广义 Rayleigh 商数。最大化 $J(\mathbf{w})$的向量 $\mathbf{w}$ 必须满足如下条件：

$$\mathbf{C}_b\mathbf{w} = \lambda\mathbf{C}_t\mathbf{w} \tag{4.78}$$

式(4.78)是一个广义特征值问题，认识到在我们的情况中矩阵积 $\mathbf{C}_b\mathbf{w}$ 总是沿向量差 $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$ 的方向，我们发现式(4.78)的解为

$$\mathbf{w} = \mathbf{C}_t^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \tag{4.79}$$

该解称为 Fisher 的线性判别式(Duda and Hart，1973)。

回到特征检测的问题，回忆式(4.74)的判别函数②和模式变换到网络隐藏层空间的类间协方差矩阵及总体协方差矩阵有关。判别函数②起着与 Fisher 线性判别式相同的作用，这就是为什么神经网络可以非常好的执行模式分类任务的理由。

4.10 反向传播和微分

反向传播是用于在多层前馈网络的权值空间中实现梯度下降的一种特殊技巧。其基本思想是有效计算一个近似函数 $F(\mathbf{w},\mathbf{x})$的偏导数，对于给定输入向量 $\mathbf{x}$ 的值近似函数 $F(\mathbf{w},\mathbf{x})$由网络根据可调整权值向量 $\mathbf{w}$ 的所有元素实现。这一点决定了反向传播算法的计算能力[5]。

进一步，假定一个多层感应器有一个 m_0 个节点的输入层，两个隐藏层，以及一个单一的输出神经元，如图 4-18 所示。

202

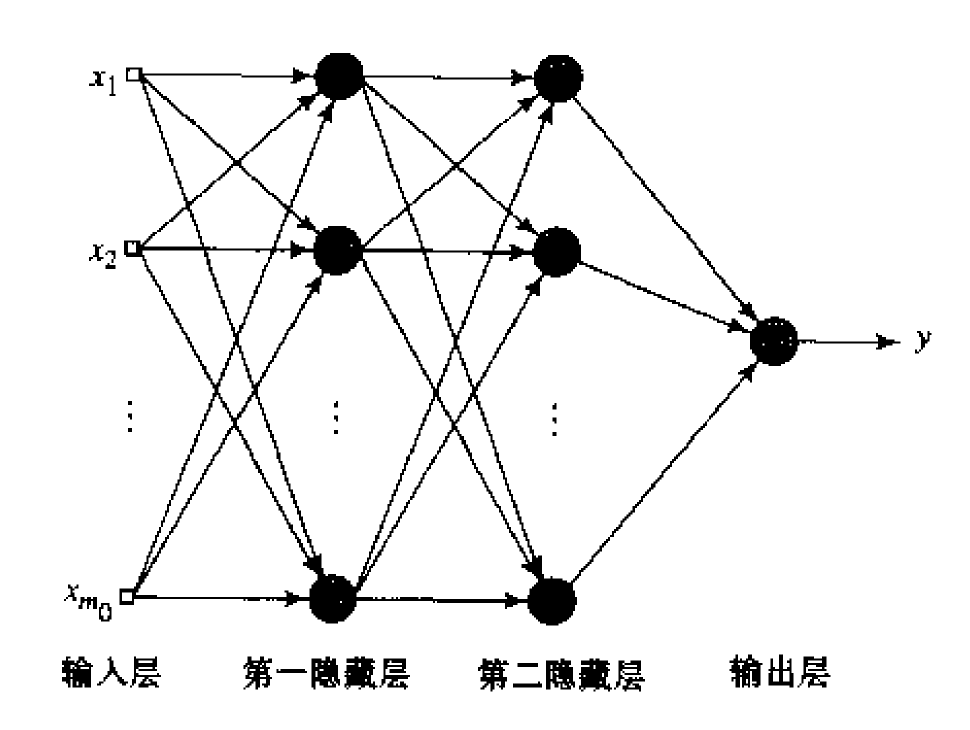


图 4-18 具有两个隐藏层和一个输出层的多层感知器

权值向量 **w** 的元素根据层数(从第一个隐藏层开始)然后根据层内的神经元和最后根据神经元中突触的数目来排序。令 $w_{ji}^{(l)}$ 表示从神经元 i 到层 $l=0$, 1, 2, …中的神经元 j 的突触权值。对于 $l=1$, 对应于第一个隐藏层, 序号 i 表示一个源结点而不是一个神经元; 对于 $l=3$, 对应于图 4-18 的输出层, 我们有 $j=1$。对于一个特定的输入向量 $\mathbf{x}=[x_1, x_2, \cdots, x_{m_0}]^T$, 我们希望计算函数 $F(\mathbf{w},\mathbf{x})$对向量 **w** 的所有元素的导数值。注意对于 $l=2$(即第二个隐藏层), 函数 $F(\mathbf{w},\mathbf{x})$具有类似于式(4.69)右边的形式。我们包含权值向量 **w** 作为函数 F 的变量, 并将注意力放在其上。

图 4-18 的多层感知器被结构$\mathscr{A}$(表示一个离散参数)和一个权值向量 **w**(由连续的元素组成)参数化。令$\mathscr{A}_j^{(l)}$ 表示从输入层($l=0$)到层 $l=1$, 2, 3 内的节点 j 所扩展成的部分结构。因此, 我们可以写成

$$F(\mathbf{w},\mathbf{x}) = \varphi(\mathscr{A}_1^{(3)}) \tag{4.80}$$

这里 φ 是激活函数。然而, $\mathscr{A}_1^{(3)}$ 仅仅被认为是一个结构符号而不是一个变量, 因此, 改写式(4.1)、(4.2)、(4.11)和(4.23)使之在这种情况下可用, 我们得到如下结果:

$$\frac{\partial F(\mathbf{w},\ \mathbf{x})}{\partial w_{1k}^{(3)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi(\mathscr{A}_k^{(2)}) \tag{4.81}$$

$$\frac{\partial F(\mathbf{w},\ \mathbf{x})}{\partial w_{kj}^{(2)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi'(\mathscr{A}_k^{(2)})\varphi(\mathscr{A}_j^{(1)})w_{1k}^{(3)} \tag{4.82}$$

$$\frac{\partial F(\mathbf{w},\mathbf{x})}{\partial w_{ji}^{(1)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi'(\mathscr{A}_j^{(1)})\mathbf{x}_i\Big[\sum_k w_{1k}^{(3)}\varphi'(\mathscr{A}_k^{(2)})w_{kj}^{(2)}\Big] \tag{4.83}$$

这里 φ'是非线性 φ 关于其输入的偏导数, x_i 是输入向量 **x** 的第 i 个元素。用相似的方法我们可以得到一般的具有更多的隐藏层和在输出层上有更多神经元的网络的偏导等式。

式(4.81)至(4.83)对于计算网络函数 $F(\mathbf{w},\mathbf{x})$关于权值向量 **w** 的元素变化的灵敏度提供了基础。令 ω 表示权值向量 **w** 的元素, $F(\mathbf{w},\mathbf{x})$关于 ω 的灵敏度定义为

[203]

$$S_\omega^F = \frac{\partial F/F}{\partial\omega/\omega}, \qquad \omega \in \mathbf{w}$$

由于这个原因我们把图 4-7 中信号流图的较低部分称为"灵敏度图"。

Jacobi 矩阵

令 W 表示一个多层感知器自由参数(即突触权值和偏置)的总数, 参数按形成权值向量 **w** 的方式排序。令 N 表示用于训练网络的样本总数。对于训练集中的给定样本 $\mathbf{x}(n)$, 利用反向传播我们可以计算近似函数 $F[\mathbf{w},\mathbf{x}(n)]$对权值向量 **w** 元素的偏导数。对于 $n=1$, 2, …, N 重复上述计算, 最后得到一个 $N\times W$ 的偏导数矩阵。这个矩阵被称为多层感知器的在 $\mathbf{x}(n)$处 Jacobi 矩阵 **J**。Jacobi 矩阵每列对应于训练集中的一个样本。

实验证据显示许多神经网络训练问题是内在"病态的", 导致 Jacobi 矩阵 **J** 几乎总是秩亏损的(Saarinen et.al., 1991)。矩阵的秩是矩阵的列或行的线性无关组的数目中最小的一个。假如秩小于 $\min(N,W)$, 我们说 Jacobi 矩阵 **J** 是秩亏损的。在 Jacobi 矩阵中任何的秩亏损导致反向传播算法仅仅得到可能搜寻方向上的部分信息, 从而导致训练时间过长。

4.11 Hessian 矩阵

代价函数$\mathscr{E}_{av}(\mathbf{w})$的 Hessian 矩阵用 **H** 表示, 定义为$\mathscr{E}_{av}(\mathbf{w})$对权值向量 **w** 的二阶导数, 显

示为

$$\mathbf{H} = \frac{\partial^2 \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}^2} \tag{4.84}$$

Hessian 矩阵在研究神经网络中起着重要作用；我们尤其要提出以下几点[6]：

1.Hessian 矩阵的特征值对反向传播学习动力学有着深远的影响；

2.Hessian 矩阵的逆为从一个多层感知器中修剪(即删除)不重要的突触权值提供基础，如 4.15 节所讨论；

3.Hessian 矩阵是形成二阶优化方法的基础，二阶优化方法可作为反向传播学习的替代，如 4.18 节所讨论。

在 4.15 节给出一个计算 Hessian 矩阵的迭代程序[7]，在本节中我们将注意放在点 1。

在第 3 章我们说明了 Hessian 矩阵的特征结构对 LMS 算法的收敛性质有重大影响。它对 [204] 反向传播算法也一样，但是更为复杂。典型地用反向传播算法来训练的多层感知器其误差曲面的 Hessian 矩阵有如下的特征值组合(LeCun, et al., 1991; LeCun, 1993)：

- 小特征值的数目很少；
- 中等大小的特征值的数目很多；
- 大特征值的数目很少。

影响这个组合的因素可分组如下：

- 非零均值的输入信号或非零均值的神经元诱导输出信号。
- 输入信号向量的元素之间的相关性和神经元诱导输出信号之间的相关性。
- 代价函数对于网络中神经元突触权值的二阶导数随着我们从一层到下一层进行处理有很宽的变化范围。在较低的层中二阶导数通常更小，这样突触权值在第一隐藏层的学习很慢，但在最后一层就学习较快。

从第 3 章我们可以回忆起 LMS 算法的学习时间对条件数 $\lambda_{max}/\lambda_{min}$ 的变化很灵敏，这里 λ_{max} 是 Hessian 矩阵最大的特征值，而 λ_{min} 是 Hessian 矩阵最小的非 0 特征值。实验结果显示反向传播算法有着相同的结果，反向传播算法是 LMS 算法的一个推广。对于非零均值的输入，它的比值 $\lambda_{max}/\lambda_{min}$ 比相应的零均值输入的比值要大：输入的均值越大，比值 $\lambda_{max}/\lambda_{min}$ 越大(见习题 3.10)。这个观察对反向传播学习动力学有着重要意义。

为了学习时间最小化，应避免使用非零均值的输入。现在，就考虑应用于一个多层感知器的第一隐藏层的神经元的信号向量 $\mathbf{x}$(即应用于输入层的信号向量)而论，$\mathbf{x}$ 应用于网络之前先减去它的每个元素一个平均值是很容易的。但是将信号应用到剩下的隐藏层和输出层中的神经元情况又会如何呢？这个问题的答案在于网络中使用的激活函数的类型。假如激励函数是非对称的(比如 logistic 函数)，每个神经元的输出界于[0,1]区间。这样的选择为那些位于网络中第一隐藏层之后的神经元带来了一个系统偏差源。为了克服这一问题我们需要利用一个如同双曲正切函数的反对称函数。对于后一种选择，每个神经元的输出可以是区间[-1,1]中的任何正值和负值，在这种情况下，它的均值可能为 0。假如网络连接数很大，用反对称激活函数的反向传播学习可能比一个使用非对称激活函数的相似过程有着更快的收敛，对此也被经验证明(LeCun et al., 1991)，这为 4.6 节描述的启发 3 提供合理性依据。

4.12 泛化

[205] 在反向传播学习中，我们一般从一个训练样本开始，而且通过向网络中装载(编码)尽可

能多的训练样本来使用反向传播算法计算一个多层感知器的突触权值。希望这样设计的神经网络可以泛化(推广)。对于从未在生成或训练网络时使用过的测试数据，若网络计算的输入－输出映射对它们来说是正确(或接近于正确)的，我们认为网络的泛化是很好的；术语“泛化”是从心理学中借用来的。这里假定测试数据是从用于生成训练数据的相同数据集抽取出来的。

学习过程(即神经网络的训练)可以看作是一个“曲线拟合”的问题。网络本身可以被简单地认为是一个非线性输入－输出映射。这个观点允许我们不再把神经网络的泛化看作是它的一个神秘的特性，而是作为相当简单的关于输入数据非线性插值的结果(Wieland and Leighton, 1987)。这种网络能够完成有意义的插值过程主要是因为具有连续激活函数的多层感知器导致输出函数同样也是连续的。

图 4-19a 表明一个假定的网络是如何进行泛化的。图中描绘的曲线所代表的非线性输入/输出映射是由网络通过对标有“训练数据”的点进行学习的结果来计算的。曲线上标有“泛化”的点就是由这个网络完成的插值结果。

一个神经网络设计得具有很好泛化能力，即使在输入与训练网络的样本稍有不同的情况下它也能够产生一个正确的输入/输出映射，这正如图中所显示的一样。然而，当一个神经网络对太多的样本进行学习的时候，它可能会完成对训练数据的记忆。这种情况可能会出现在找到一个存在于训练数据中但对于将要建模的固有函数却为假的特征(例如，由于噪声)的时候。这种现象称为“过拟合”或者“过训练”。当网络被过训练的时候，它就失去了在相近输入/输出模式之间进行泛化的能力。

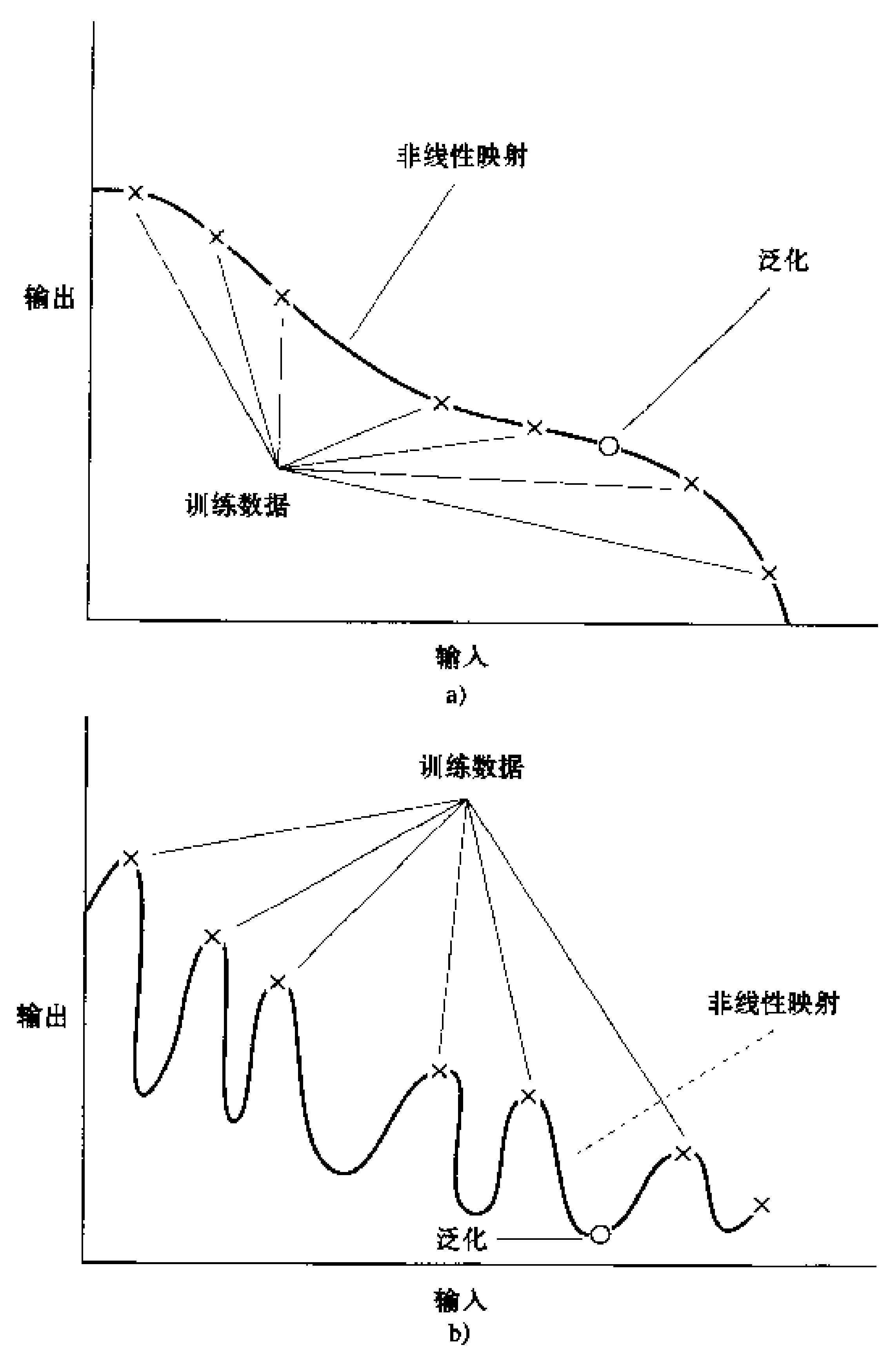


图 4-19
a)恰当地拟合数据(良好泛化) b)过拟合数据(差的泛化)

通常，用这种方法把数据装载到多层感知器要求使用比实际需要更多的隐藏层神经元，结果导致在网络的突触权值中存储了输入空间中由于噪声引起的非期望因素。例如，在图 4-19a 相同的数据条件下，图 4-19b 显示由于神经网络中的记忆导致泛化不佳是如何出现的

例子。“记忆”本质上是一个“查询表”，这意味着由神经网络计算的输入/输出映射是非光滑的。正如在 Poggio and Girosi(1990a)文章中指出的那样，输入/输出映射的光滑性和如 Occam 剃刀(Occam's razor)之类的模型选择标准紧密相关，在没有相反的先验知识情况下它的核心本质是选择“最简单”函数。针对于我们给出的讨论，最简单函数是指在给定的误差标准下逼近一个给定映射的函数中最光滑的函数，因为这个选择总体上要求最少的计算资源。依赖于研究现象的规模范围，光滑性在许多应用上同样是自然的。因而为不适定的输入/输出关系寻找一个光滑的非线性映射是重要的，使得网络能够根据训练模式将新模式正确地分类(Wieland and Leighton,1987)。

206

为有效的泛化给出充分的训练集大小

下面的三个因素是对泛化产生影响：(1)训练集的大小，以及它如何表示感兴趣的环境；(2)神经网络的体系结构；(3)当前问题的物理复杂度。无疑地，我们无法对后者进行控制。在另外的两个因素中，我们可以从两个不同的方面考察泛化问题(Hush and Horne,1993)：

- 网络的体系结构是固定的(可期望与固有问题的物理复杂度一致)，需要解决的问题是决定一个产生好的泛化必须的训练集的大小。
- 训练集的大小是固定的，感兴趣的问题是决定最好的网络体系结构使得具有好的泛化。

207

在它们各自的方法里这两种观点都是合理的。当前我们集中讨论第一种观点。

适度的训练样本大小或样本复杂度问题已经在第2章中讨论过了。正如在该章中指出的那样，VC 维数为这个重要的设计问题的原则性解决方法提供了理论基础。特别地，我们有与分布无关和最坏情形下的公式以估算能够足够形成一个好的泛化性能的训练样本的大小；请参见 2.14 节。不幸的是，我们经常发现在实际需要的训练样本的大小和由这些公式预测的训练样本的大小之间存在着巨大的数值差异。正是这个差异使得样本复杂度问题成为一个持续公开的研究领域。

在实践中，看来一个好的泛化事实上我们所需要的全部是训练集的大小 N 满足条件

$$N = O\left(\frac{W}{\varepsilon}\right) \tag{4.85}$$

在这里 W 是指网络中自由参数(即突触权值和偏置)的总数，ε 表示测试数据中容许分类误差的部分(正如在模式分类中一样)。$O(\cdot)$表示所包含的量的阶数。例如，具有 10%误差的所需训练样本数量应该是网络中自由参量数量的 10 倍。

式(4.85)与用于 LMS 算法的 Widrow 经验方法是一致的，后者指出线性自适应时间滤波的适应迟滞时间近似等于一个自适应抽头延迟线滤波器的记忆范围除以误调节(Widrow and Stearns,1985)。LMS 算法中的误调节扮演的角色与式(4.85)中的误差 ε 有某些相似。这个经验规则的进一步理由将在下一节中介绍。

4.13 函数逼近

一个由反向传播算法训练的多层感知器可以被看作一个实现一般性质的非线性输入/输出映射的实际工具。具体地，令 m_0 表示多层感知器的输入(源)节点的数目，令 $M = m_L$ 表示网络中输出层神经元的数目。网络的输入/输出关系定义一个从 m_0 维欧几里德输入空间

到 M 维欧几里德输出空间的映射，当激活函数是无限连续可微的时候，这个映射也是无限连续可微的。在用这种输入/输出映射观点来评价多层感知器能力的过程中，提出了下面基本的问题：

一个多层感知器的输入/输出映射能够提供任何一个连续映射的近似实现，它的隐藏层层数的最小数目是多少？

通用逼近定理

这个问题可以用一个非线性输入/输出映射的通用逼近定理[8]来具体表达，该定理陈述如下：

令 $\varphi(\cdot)$ 是一个非常数的、有界的和单调增的连续函数。令 I_{m_0} 表示 m_0 维单位超立方体 $[0,1]^{m_0}$。I_{m_0} 上连续函数空间用 $C(I_{m_0})$ 表示。那么，给定任何函数 $f \ni C(I_{m_0})$ 和 $\varepsilon > 0$，存在这样的一个整数 m_1 和实常数 α_i，b_i 和 w_{ij}，其中 $i = 1, \cdots, m_1$，$j = 1, \cdots, m_0$，使我们可以定义

$$F(x_1, \cdots, x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \varphi\left(\sum_{j=1}^{m_0} w_{ij} x_j + b_i \right) \tag{4.86}$$

208

作为 $f(\cdot)$ 函数的一个近似实现；也就是说，

$$\left| F(x_1, \cdots, x_{m_0}) - f(x_1, \cdots, x_{m_0}) \right| < \varepsilon$$

对存在于输入空间中的所有 x_1，x_2，$\cdots$，x_{m_0} 均成立。

通用逼近定理可直接用于多层感知器。我们首先注意到在一个作为多层感知器结构的神经元模型中作为非线性部分的 logistic 函数 $1/[1+\exp(-v)]$ 是一个真正非常数的、有界的和单调递增的函数；因此它满足函数 $\varphi(\cdot)$ 的上述条件。下一步，我们注意式(4.86)表达如下所述的多层感知器的输出：

1. 网络具有 m_0 个输入节点和单个由 m_1 个神经元组成的隐藏层；输入由 x_1，$\cdots$，x_{m_0} 表示。

2. 隐藏神经元 i 具有突触权值 w_{i_1}，$\cdots$，w_{m_0}，偏置 b_i。

3. 网络的输出是隐藏层的线性组合，带有定义输出层突触权值的 α_1，$\cdots$，α_{m_1}。

通用逼近定理是存在性定理，它与精确表示相反，为任意连续函数的逼近提供数学上的基础。作为定理的本质，式(4.86)仅仅是推广有限 Fourier 级数逼近。事实上，这个定理说明，对于多层感知器计算一个由输入 x_1，$\cdots$，x_{m_0} 和期望(目标)输出 $f(x_1, \cdots, x_{m_0})$ 表示的给定训练集的一致 ε 逼近来说，单个隐藏层是足够的。然而，定理并没有说明单个隐藏层在学习时间、实现的难易程度或者(更重要的)泛化意义上是最优的。

逼近误差的界

假定网络使用 sigmoid 函数的单层隐藏神经元和线性输出神经元，Barron(1993)建立了多层感知器的逼近性质。网络通过使用反向传播算法训练，然后用新的数据测试。在训练过程

中，网络根据训练数据学习目标函数 f 中的特殊点，从而产生由式(4.86)中定义的逼近函数 F。当网络遇到以前没有见过的测试数据的时候，网络函数 F 就充当目标函数中新的点的估计器；即，$F=\hat{f}$。

一个目标函数的光滑度属性用它的 Fourier(变换)来表达。特别地，用 Fourier 幅度分布加权后的频率向量的范数的平均值作为函数 f 振荡的度量标准。令 $\tilde{f}(\boldsymbol{\omega})$ 表示函数 $f(\mathbf{x})$ 的多维 Fourier 变换，$\mathbf{x}\in\mathbb{R}^{m_0}$；$m_0\times 1$ 向量 $\boldsymbol{\omega}$ 为频率向量。函数 $f(x)$ 由关于它的 Fourier 变换函数 $\tilde{f}(\boldsymbol{\omega})$ 的反变换公式定义如下：

$$f(x)=\int_{\mathbb{R}^{m_0}}\tilde{f}(\boldsymbol{\omega})\exp(j\boldsymbol{\omega}^T\mathbf{x})\,d\boldsymbol{\omega} \tag{4.87}$$

在这里 $j=\sqrt{-1}$。对于复值函数 $\tilde{f}(\boldsymbol{\omega})$，由于 $\boldsymbol{\omega}\tilde{f}(\boldsymbol{\omega})$ 是可积的，我们定义函数 f 的 Fourier 幅度分布的一阶绝对动量如下：

$$C_f=\int_{\mathbb{R}^{m_0}}|\tilde{f}(\boldsymbol{\omega})|\times\|\boldsymbol{\omega}\|^{1/2}\,d\boldsymbol{\omega} \tag{4.88}$$

其中 $\|\boldsymbol{\omega}\|$ 为 $\boldsymbol{\omega}$ 的欧几里德范数，$|\tilde{f}(\boldsymbol{\omega})|$ 为 $\tilde{f}(\boldsymbol{\omega})$ 的绝对值。一阶绝对动量 C_f 量化函数 f 的光滑度或正则性。

一阶绝对动量 C_f 为使用以式(4.86)中输入/输出映射函数 $F(\mathbf{x})$ 为表示的多层感知器近似 $f(\mathbf{x})$ 而导致的误差范围的界提供基础。近似误差可以用与一个半径 $r>0$ 的球体 $B_r=\{\mathbf{x}:\|\mathbf{x}\|\leqslant r\}$ 中任意可能的概率测度 μ 相关的积分平方误差来衡量。在这个基础上我们可以对 Barron(1993)提出的近似误差范围的界提出如下命题：

对于每个具有有限一阶绝对动量 C_f 的连续函数 $f(\mathbf{x})$，以及每个 $m_1\geqslant 1$，存在一个由式(4.86)定义的 sigmoid 函数的线性组合 $F(\mathbf{x})$，使得

$$\int_{B_r}(f(\mathbf{x})-F(\mathbf{x}))^2\mu(d\mathbf{x})\leqslant\frac{C'_f}{m_1}$$

其中 $C'_f=(2rC_f)^2$。

当在严格属于球体 B_r 内部的输入向量 $\mathbf{x}$ 的值集合 $\{\mathbf{x}_i\}_{i=1}^N$ 上观察函数 $f(\mathbf{x})$ 的时候，命题的结果对经验风险提供如下的界：

$$R=\frac{1}{N}\sum_{i=1}^{N}(f(\mathbf{x}_i)-F(\mathbf{x}_i))^2\leqslant\frac{C'_f}{m_1} \tag{4.89}$$

在 Barron(1992)中，利用式(4.89)的逼近结果表示使用具有 m_0 个输入节点和 m_1 个隐藏神经元的多层感知器而导致的风险 R 的界如下：

$$R\leqslant O\left(\frac{C_f^2}{m_1}\right)+O\left(\frac{m_0m_1}{N}\log N\right) \tag{4.90}$$

风险 R 的界中的两项表达两种对隐藏层大小互相冲突的要求之间的折衷：

1. 最佳逼近的精确度。为了满足这个要求，根据通用逼近定理隐藏层的大小 m_1 必须足够大；

2. 近似的经验拟合精确度。为了满足的第二个要求，我们必须使用一个小的比值 m_1/N。由于训练集的固定大小为 N，隐藏层的大小 m_1 应该保持较小，这跟第一个要求是矛盾的。

式(4.90)描述的风险 R 的界具有另外一个有趣的含意。特别地，我们看到假如一阶绝对动量 C_f 仍是有限的话，相对于输入空间维数 m_0 一个指数规模的大样本集对于得到一个目标函数精确的估算并不是必须的。这个结果使得多层感知器作为通用逼近器在实际条件下甚至显得更重要。

经验拟合和最佳逼近之间的误差可以看作是第 2 章中所述的估计误差。令 ε_0 表示估计误差的均方值。然后忽略式(4.90)中表达式的第二项的对数因子 $\log N$，我们可以推断出一个好的泛化所需的训练集大小 N 大约是 $m_0 m_1/\varepsilon_0$。这个结果跟经验公式(4.85)具有相似的数学结构，记住 $m_0 m_1$ 等于网络中自由参数 W 的总数。换句话说，我们可以从总体上说为了得到好的泛化，训练样本的数目 N 应该大于网络中自由参数总数和估计误差均方值之比。

维数灾

出现在式(4.90)所描述的界中另一个有趣的结果，是当对隐藏层的大小通过设定

$$m_1 \simeq C_f\left(\frac{N}{m_0 \log N}\right)^{1/2}$$

进行优化(也就是风险 R 关于 N 最小化)的时候，这时风险 R 由 $O(C_f\sqrt{m_0(\log N/N)})$ 限定。这个结果的一个令人惊奇的方面是根据风险 R 的一阶行为，以训练集大小 N 的函数表达的收敛速率的阶为 $(1/N)^{1/2}$(乘以一个对数因子)。在另一方面，对传统的光滑函数(例如多项式和三角函数)我们有不同的行为。令 s 表示光滑度的一种度量，定义为函数具有连续导数的阶数。那么，对于传统光滑函数我们发现总风险 R 的极小极大的收敛速率的阶为 $(1/N)^{2s/(2s+m_0)}$。这个收敛速率对输入空间维数 m_0 的依赖就是维数灾，这严重地制约这些函数的实际应用。使用多层感知器进行函数逼近看来提供超越于传统光滑函数的优势；但是，这个优势受限于一阶绝对动量 C_f 保持有限的条件；这是一个光滑度约束。

Richard Bellman 在他对自适应控制过程(Bellman, 1961)的研究中介绍了维数灾。为了从几何上解释这个概念，令 $\mathbf{x}$ 表示一个 m_0 维的输入向量，$\{(\mathbf{x}_i, d_i)\}, i = 1, 2, \cdots, N$ 表示训练样本。采样密度与 N^{1/m_0} 成正比。令函数 $f(\mathbf{x})$ 代表一个位于 m_0 维输入空间的曲面，它近似通过点 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$。现在，如果函数 $f(\mathbf{x})$ 是任意复杂并且(对绝大部分)是完全未知的，我们需要密集的样本(数据)来进行很好的学习。不幸的是，密集样本在“高维”中是很难找到的，因此产生了维数灾。特别地，维数增加的结果导致复杂度呈指数增长，从而引起高维空间中一致随机分布点的空间填充性质退化。维数灾的基本原因如下(Friedman, 1995)：

[211]

定义在高维空间的函数很可能远远比定义在低维空间上的函数复杂得多，并且这些复杂的东西是更难以区分的。

克服维数灾难的惟一可行办法是除训练数据外结合关于这个函数的一些先验知识，这些先验知识已知是正确的。

在实际中，也可能存在这样的争论：如果希望在高维空间中得到好的估计，随着输入维数的增加我们必须增加未知的固有函数的光滑度(Niyogi and Girosi, 1996)。这个观点将在第 5 章中继续深入讨论。

可行性考虑

从理论的观点来看，通用逼近定理是重要的，因为它为具有单个隐藏层的前馈网络作为一类逼近器的可能性提供了必要的数学工具；如果没有这样一个理论，我们可能在盲目寻找那些并不存在的方法。然而，这个理论并不是构造性的，即它实际上并不能具体实现如何由陈述的逼近性质决定一个多层感知器。

通用逼近定理假设被逼近的连续函数是给定的并且逼近可用一个神经元数目无限制的隐藏层。这两个假设在多层感知器的绝大多数实际应用中都是不满足的。

使用单个隐藏层的多层感知器的问题是隐藏层的神经元倾向于全局地相互作用。在复杂情形下这种相互作用使得在一点提高它的逼近同时又很难不恶化它在另外点上的逼近。另一方面，在具有两个隐藏层的情况下逼近(曲线拟合)过程变得更容易协调。具体地，我们可以进行如下处理(Funahashi,1989;Chester,1990)：

1. 从第一个隐藏层中抽取局部特征。特别地，利用在第一个隐藏层中的一些神经元将输入空间分割成区域，这层中另外的神经元学习表征这些区域特点的局部特征。

2. 从第二个隐藏层中抽取全局特征。特别地，在第二隐藏层中的一个神经元组合在输入空间特定区域操作的第一个隐藏层的各神经元的输出，从而学习该区域的全局特征并且在别处的输出为零。

这个两阶段的逼近过程在实质上与曲线拟合的样条插值技术是相似的，相似的意义是指神经元的作用效果是分离的且输入空间不同区域的逼近可以单独地调整。一个样条就是一个分段多项式逼近的例子。

212

Sontag(1992)为在逆问题中两个隐藏层的使用提供进一步理由。具体地，考虑下述逆问题：

给定一个连续向量值的函数 $\mathbf{f}: \mathbb{R}^m \to \mathbb{R}^M$，一个紧子集 $\mathscr{C} \subseteq \mathbb{R}^M$ 包含在 $\mathbf{f}$ 的像(即值域)之中，并且 $\varepsilon > 0$，寻找一个向量值函数 $\varphi: \mathbb{R}^M \to \mathbb{R}^m$，使得满足下述条件：

$$\| \varphi(\mathbf{f}(\mathbf{u})) - \mathbf{u} \| < \varepsilon \qquad 对于\ \mathbf{u} \in \mathscr{C}$$

这个问题出现在逆运动学(动力学)中，此时一个系统的观察状态 $\mathbf{x}(n)$ 是当前动作 $\mathbf{u}(n)$ 和系统前一状态 $\mathbf{x}(n-1)$ 的函数，表示为

$$\mathbf{x}(n) = \mathbf{f}(\mathbf{x}(n-1), \mathbf{u}(n))$$

假设 $\mathbf{f}$ 可逆，使得对于任何 $\mathbf{x}(n-1)$ 我们可以把 $\mathbf{u}(n)$ 当作 $\mathbf{x}(n)$ 的函数来求解。函数 $\mathbf{f}$ 代表直接运动学，因而函数 φ 代表逆运动学。在实际条件中，我们的动机是寻找一个可以通过多层感知器计算的函数 φ。从总的说来，不连续函数 φ 对于解决逆运动学问题是必需的。有趣的是即使允许使用具有不连续激活函数的神经元模型，一个隐藏层并不能充分保证所有这类逆问题的解决，但是具有两个隐藏层的多层感知器对于每一个可能的 $\mathbf{f}$、$\mathscr{C}$ 和 ε 是充分的(Sontag,1992)。

4.14 交叉确认

反向传播学习的本质是把输入/输出映射(由标定的一组训练样本表示)编码为一个多层感知器的突触权值和阈值。希望网络被很好地训练使得它对过去进行充分的学习就能对未来进行泛化。从这个观点来看，学习过程意味着对这个数据集合给出网络参数化的一个选择。具体地，我们可以把网络选择问题看作是在一组候选模型结构(参数)集合中选择符合某个标

准的“最好”的一个。

在这种意义下，统计学中一个名为交叉确认的标准工具提供一个有吸引力的指导原则[9]（Stone,1974,1978）。已有的可用数据集首先被随机分割成一个训练集和一个测试集。这个训练集被进一步细分为两个不相交子集：

- 估计子集，用来选择模型。
- 确认子集，用来测试或者确认模型。

这里的动机是用一个与参数估计数据集不同的数据集确认模型。用这个办法我们可以用训练集来估计不同候选模型的性能，进而选择“最好”的一个。然而，存在一个明显的可能性是这样选出来的具有最好表现参数值的模型可能会导致对确认子集的过度拟合。为了防止这个可能性的出现，在与确认子集不同的测试集上测量被选模型的泛化性能。

当我们不得不以设计一个具有好的泛化性能的大型神经网络作为目标的时候，交叉确认的使用是特别吸引人的。例如，我们可以使用交叉确认确定具有最优隐藏神经元数目的多层感知器，以及最好在何时停止它的训练，正如在下面两小节中所述的那样。 213

模型选择

根据交叉确认选择模型的思想，遵循一种与第2章所述结构风险最小化相似的原理。现在考虑如下表示的布尔函数类的嵌入结构：

$$\mathscr{F}_1 \subset \mathscr{F}_2 \subset \cdots \subset \mathscr{F}_n$$

$$\mathscr{F}_k = \{F_k\} = \{F(\mathbf{x},\mathbf{w});\mathbf{w} \in \mathscr{W}_k\}, \qquad k = 1,2,\ldots,n \tag{4.91}$$

也就是说，第 k 个函数类$\mathscr{F}_k$ 包含一簇具有相似体系结构的多层感知器，其权值向量 $\mathbf{w}$ 从一个多维权值空间$\mathscr{W}_k$ 抽出。以函数或者假设 $F_k = F(\mathbf{x},\mathbf{w})$，$\mathbf{w}\in\mathscr{W}_k$ 为特征的类的一个成员把输入向量 $\mathbf{x}$ 映射到$\{0,1\}$，这里 $\mathbf{x}$ 是以某未知概率 P 从输入空间$\mathscr{X}$中抽取出来的。在所述结构中每个多层感知器都是由反向传播算法训练的，该算法负责多层感知器参数的训练。模型选择问题本质是选择具有最好的自由参数（即突触权值和阈值）数目 W 值的多层感知器。更精确地，假设对输入向量 $\mathbf{x}$ 的期望响应标量是 $d=\{0,1\}$，我们定义泛化误差如下：

$$\varepsilon_g(F) = P(F(\mathbf{x}) \neq d) \qquad \text{对于 } \mathbf{x} \in \mathscr{X}$$

给出一个标定的训练样本集

$$\mathscr{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$$

我们的目标是选择特定的假设 $F(\mathbf{x},\mathbf{w})$，当从测试集中给定输入时它最小化所得泛化误差 $\varepsilon_g(F)$。

下面我们假设由式(4.91)表达的结构具有这样的性质，即对于任意大小的 N 我们都可以找到一个具有数量足够多的自由参数的数目 $W_{max}(N)$ 的多层感知器，使得训练数据集$\mathscr{T}$就可以被合适地拟合。这只不过重申4.13节的通用逼近定理。我们把 $W_{max}(N)$ 称为拟合数。$W_{max}(N)$ 的意义在于，一个合理的模型选择程序应该选择一个满足 $W \leqslant W_{max}(N)$ 的假设 $F(\mathbf{x},\mathbf{w})$；否则网络复杂度将会增加。

令一个位于0和1范围之间的参数 r 决定估计子集和确认子集之间的训练数据集$\mathscr{T}$的划分。$\mathscr{T}$ 由 N 个样本组成，$(1-r)N$ 个样本分配给估计子集，剩下的 rN 个样本分配给确认子集。估计子集用$\mathscr{T}'$表示，它用于训练多层感知器的一个嵌套序列，嵌套结构导致复杂度递增 214

的假设$\mathscr{F}_1$，$\mathscr{F}_2$，…，$\mathscr{F}_n$。由于$\mathscr{T}'$由$(1-r)N$个样本组成，我们认为W的值小于或者等于相应的拟合数$W_{max}((1-r)N)$。

交叉确认方法的使用导致选择

$$\mathscr{F}_{cv} = \min_{k=1,2,\cdots,v} \{e''_t(\mathscr{F}_k)\} \tag{4.92}$$

其中v对应于$W_v \leq W_{max}((1-r)N)$，$e''_t(\mathscr{F}_k)$是在由rN个样本组成的确认子集$\mathscr{T}''$上测试时由假设$\mathscr{F}_k$产生的分类误差。

关键问题是如何具体确定参数r以决定训练集$\mathscr{T}$在估计子集$\mathscr{T}'$和确认子集$\mathscr{T}''$之间的划分。在Kearns(1996)描述的研究中，利用VC维数对该论题进行分析处理和具体的计算机仿真支持，确定了最优r的几个定性特点：

- 当定义输入向量$\mathbf{x}$的期望响应d的目标函数的复杂度相对于样本大小的N是很小的时候，交叉确认的性能对r的选择相对不灵敏。
- 随着目标函数相对于样本大小N变得更复杂的时候，最优r的选择在交叉确认性能上具有更重要的影响，并且r自身的值减小。
- r的一个单一固定的值在目标函数复杂度的一个相当大的范围内保持近乎最佳。

根据Kearns(1996)报告的结果，r等于0.2的一个固定值看来是一个合理的选择，这意味着训练集$\mathscr{T}$的80%被指定为估计子集，剩下的20%被指定为确认子集。

早些时候我们谈到复杂度增长的多层感知器的嵌入序列。对于规定的输入和输出层来说，这样的顺序是可能被建立起来的，例如，建立具有$v=p+q$个完全连接的多层感知器如下：

- p个具有隐藏神经元数目按$h'_1 < h'_2 < \cdots < h'_p$增加的单个隐藏层的多层感知器。
- q个具有两个隐藏层的多层感知器；第一个隐藏层神经元的大小为h'_p，第二个隐藏层神经元数目按$h''_1 < h''_2 < \cdots < h''_q$递增。

当我们从一个多层感知器到另一个多层感知器的时候，自由参数数目W有相应的增加。上述基于交叉确认方法的模型选择过程为我们提供一个决定多层感知器中隐藏神经元数目的原则性方法。尽管该过程针对二值分类讨论的，但是它可等价地应用到多层感知器的其他应用中。

训练的早期停止方法

通常，用反向传播算法训练的多层感知器分阶段地进行学习，随训练过程的进行从相当简单的映射函数实现到更复杂的映射函数实现。这通过在一个典型情形下在训练中均方误差随着训练回合的增加而减少的例子来证明：均方误差从一个很大的值开始，然后迅速地减小，最后随着网络在误差曲面接近局部最小值的时候缓慢地减小。由于以得到好的泛化作为目标，如果我们准备通过观察它自身训练得到的学习曲线来断定什么时候停止训练最好，这是非常困难的。特别地，根据4.12节关于泛化所说的，如果训练时间并不在恰当的点上停下来，网络结束时过拟合训练数据是可能的。

215

我们可以通过交叉确认来标记过拟合的发生，为此训练数据被分成估计子集和确认子集。使用样本的估计子集以通常方法训练网络，但有较小的修改：训练时间被周期性地停止(即每一个周期都有许多训练回合)，并且在每个训练周期之后都由确认子集测试网络。具体地，周期性的估计伴随确认(estimation-followed-by-validation)的过程是如下进行的：

- 经过一个估计(训练)周期之后，多层感知器的突触权值和偏置都已经固定，网络是在它的前向方式下运作的。从而对确认子集中的每个样本测定确认误差。
- 当确认阶段完成的时候，估计(训练)重新开始另一个周期，这个过程被重复。

这个过程称作*训练的早期停止方法*[10]。

图4-20显示两种学习曲线的概念形式，一个属于估计子集上的测定误差，另一个属于确认子集。通常，模型在确认子集上的表现并不像它在估计子集上的表现那么出色，它的设计是基于估计子集的。估计学习曲线在一般情况下随训练回合数目的增加而单调地减小。与此相对地，确认学习曲线单调地递减到一个最小值，然后它开始随训练的继续而递增。当我们仅观察估计学习曲线的时候，很明显通过越过确认学习曲线上的最小点我们可以得到它的更小的值。然而在实际上，网络在越过该点学习到的主要是包含在训练数据中的噪声。这种启发方法意味着确认学习曲线上的最小点可用于停止训练过程的合理准则。

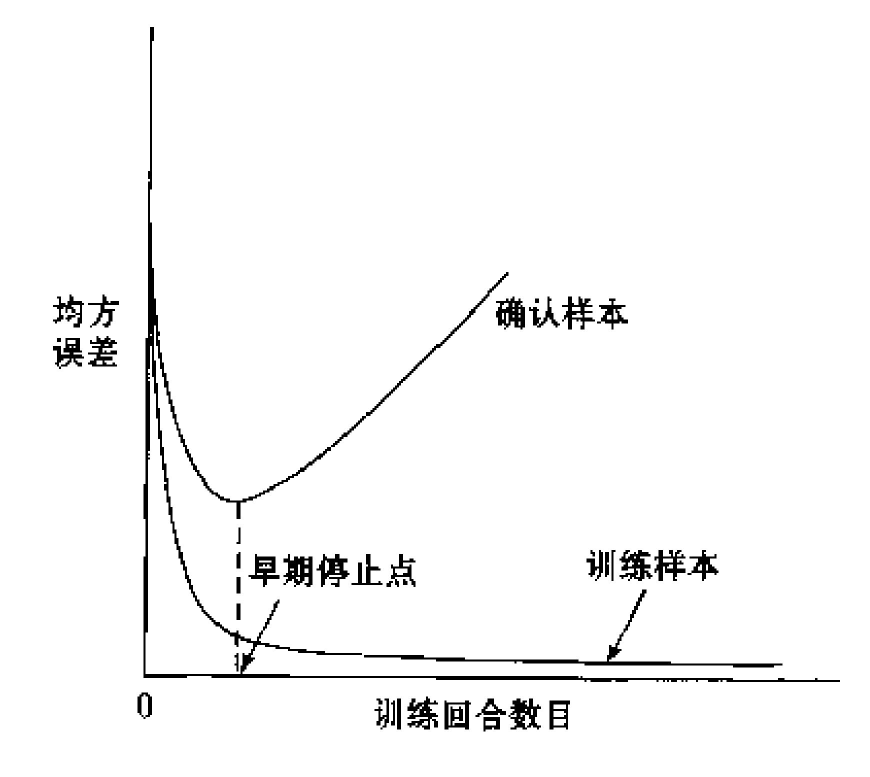


图4-20 基于交叉确认的早期停止准则示意图

216

如果训练数据是无噪声结果将会如何？我们如何为一个确定的情况判断它的早期停止？这种情况的部分答案是，如果估计和确认误差两者都不能同时地趋于零，这暗示着网络并没有建立函数的精确模型的能力。在这种情形下我们所能做到的最好事情是力求最小化误差，例如积分平方误差，它(大体上)等价于最小化通常的具有均匀输入密度的全局均方误差。

在 Amari et al.(1996)提出的过拟合现象的统计学理论为训练早期停止方法的使用提出了警告。这个理论是基于集中式学习的，并且得到包含一个隐藏层的多层感知分类器的具体计算机仿真的支持。两种行为模式同样依赖于训练集的大小：

一种是*非渐近模式*，这种模式的 $N < W$，其中 N 是训练集的大小，W 是网络中自由参数的个数。对于这种行为模式来说，训练的早期停止方法通过无遗漏训练(即用完整的样本集合进行训练并且训练过程不被停止)确实提高网络的泛化性能。这个结果提示当 $N < 30W$ 的时候过拟合可能会发生，并且交叉确认停止训练的方法的运用具有实际的优点。决定估计子集和确认子集之间训练数据划分的参数 r 的最优值定义为

$$r_{opt} = 1 - \frac{\sqrt{2W-1}-1}{2(W-1)}$$

对于大的 W，这个公式近似为

$$r_{opt} \simeq 1 - \frac{1}{\sqrt{2W}}, W\text{很大} \tag{4.93}$$

例如，对于 $W = 100$，$r_{opt} = 0.07$，这意味着训练数据的93%被分配到估计子集，而剩下的7%被分配到确认子集。

另一种是*渐近模式*，这种模式的 $N > 30W$。对于这种行为模式来说，通过无遗漏训练使用训练早期停止方法产生的泛化性能的提高是很小的。换句话说，在训练样本的大小相对大于网络参数的数目的时候，无遗漏学习是令人满意的。

交叉确认的变体

上述交叉确认的方法称为坚持到底方法(hold out method)。在实际中还有另外一些能找到它们自身应用的交叉确认的变体，特别是在标定样本缺乏的时候。在这样的情况下我们可以通过把 N 个样本的可用集合分割为 K 个子集来使用多重交叉确认方法，$K>1$；这里假设 N 对 K 是可除的。这个模型在除了一个子集之外的其他子集上进行训练，确认误差通过剩下子集上的测试来测量。这个过程总共被重复 K 次试验，每次使用一个不同的子集进行确认，如图 4-21 所示 $K=4$ 的情形。模型性能的评估是通过求实验中所有的实验的确认平方误差的平均值来进行的。多重交叉确认存在一个缺点：因为模型必须训练 K 次，它可能需要一个过多的计算量，这里 $1<K\leq N$。

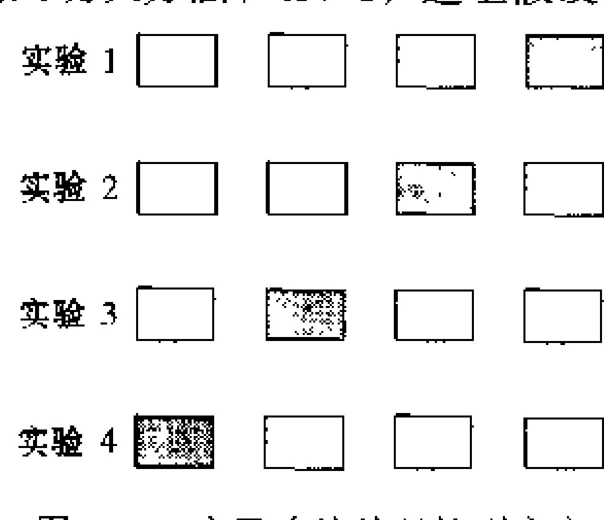


图 4-21 交叉确认的坚持到底方法示意图。对一给定的实验，带阴影的数据集用来确认模型，而剩下的数据用来训练模型

当可用的标定样本的数目 N 被严格限制的时候，我们可以使用被称为"留一"方法的多重交叉确认的极端形式。在这种方法中，$N-1$ 个样本用来训练模型，并且这个模型通过剩下的一个样本的测试来确认。这个实验总共被重复 N 次，每次留出一个不同的样本来进行确认。然后通过确认的平方误差在 N 次实验上求平均。

4.15 网络修剪技术

用神经网络解决现实世界中的问题经常要求使用一个相当庞大的高度结构化的网络。在此背景下出现的一个实际问题是在保持良好性能的同时使网络的规模最小化。具有最小规模的神经网络学习训练数据的独有特征或者噪音的可能性更小，这样可能对新的数据有更好的泛化。我们可以用如下两个途径中的一个来达到这个设计目标：

- 网络生长，在这种方法中我们以一个小的多层感知器开始，小到能实现当前任务即可，然后仅当用这个多层感知器不能实现我们具体的设计要求的时候增加一个新的隐藏神经元或者一层新的隐藏神经元[11]。
- 网络修剪，用这种方法我们以一个很大的具有足够解决当前问题性能的多层感知器开始，然后通过选择的和有序的方式削弱或者消除某些突触权值来修剪多层感知器。

在本节中我们集中讨论网络修剪的方法。特别地，我们描述两种逼近，一种基于"正则化"的形式，另一种基于从网络中"删除"某些连接的形式。

复杂性-正则化

无论用何种方式设计一个多层感知器，实际上我们都是对生成用于训练网络的输入输出样本的物理现象建立一个非线性模型。就网络的设计而论在本质上还是统计的，我们需要在训练数据的可靠性和模型的适应度之间寻找一个适当的折中(即解决偏置方差困境的方法)。在反向传播学习的背景下，或者任何其他的监督学习过程而言，我们都可能通过最小化表述如下的总量风险以实现折中：

$$R(\mathbf{w}) = \mathscr{E}_s(\mathbf{W}) + \lambda\mathscr{E}_c(\mathbf{w}) \qquad (4.94)$$

第一项$\mathscr{E}_s(\mathbf{w})$是标准的性能度量，它同时依赖于网络(模型)和输入数据。在反向传播学习中，它被典型地定义为均方误差，该误差的计算扩展到网络输出神经元，并且它在每一回合的基础上对所有训练样本来完成。第二项$\mathscr{E}_c(\mathbf{w})$是复杂性惩罚，它单独依赖于网络(模型)；它所包含的内容利用我们可能具有的关于所考虑模型的解的先验知识。事实上，式(4.94)所定义的总量风险形式是 Tikhonov 正则化理论的简单陈述；这个主题将在第 5 章详细论述。对于当前的讨论，把 λ 看作正则化参数就足够了，它代表着复杂性惩罚项关于性能度量项的相对重要性。当 λ 为零的时候，反向传播学习过程是无约束的，网络由训练样本完全确定。在另一方面，当 λ 趋于无穷大的时候，这意味着由复杂性惩罚所得到的约束自身就可以具体确定网络，用另一种说法就是训练样本是不可靠的。在权值衰减过程的实际应用中，正则化参数 λ 被赋予两个极端情形之间的某个位置的值。这里所讲述的使用复杂性正则化提高归纳能力的观点是完全和第 2 章中讨论的结构风险最小化过程相容的。

在一般设置中，复杂度惩罚项$\mathscr{E}_c(\mathbf{w})$的一个选择是第 k 阶光滑积分

$$\mathscr{E}_c(\mathbf{w},k) = \frac{1}{2}\int\left\|\frac{\partial^k}{\partial \mathbf{x}^k}F(\mathbf{x},\mathbf{w})\right\|^2\mu(\mathbf{x})\,d\mathbf{x} \tag{4.95}$$

这里 $F(\mathbf{x},\mathbf{w})$是模型实现的输入输出映射，$\mu(\mathbf{x})$ 是某个加权函数，它决定在这个输入空间中要求函数 $F(\mathbf{x},\mathbf{w})$光滑的区域。这里的目标是使得 $F(\mathbf{x},\mathbf{w})$对输入向量 $\mathbf{x}$ 第 k 阶微分较小。我们选择 k 越大，函数 $F(\mathbf{x},\mathbf{w})$就变得越光滑(即更少的复杂度)。

下面我们描述多层感知器的三种不同(难度递增)的复杂性正则化方法。 219

权值衰减 在权值衰减过程(Hinton, 1989)中，复杂性惩罚项被定义为网络中权值向量 $\mathbf{w}$(即所有的自由参数)的平方范数，表示为

$$\mathscr{E}_c(\mathbf{w}) = \|\mathbf{w}\|^2 = \sum_{i\in\mathscr{C}_{\text{total}}} w_i^2 \tag{4.96}$$

其中集合$\mathscr{C}_{\text{total}}$是指网络中所有的突触权值。这个过程是通过强迫网络中的一些突触权值取近似于零的值来进行的，而允许其他的权值保持它们相对大的值。所以，网络的权值大致分为两个类：那些对网络(模型)具有很大影响的权值和那些对网络很少或者根本没有影响的权值。在后一类中的权值称为*多余权值*。在不进行复杂性正则化的情况下，这些权值通过它们很可能取完全任意的数值，或为了得到训练误差上的轻微减少而促使网络过度拟合训练数据，从而导致很差的推广性能(Hush and Horne, 1993)。复杂性正则化的使用鼓励多余权值取得接近于零的数值，因而提高泛化能力。

在权值衰减过程中，多层感知器中所有的权值都被平等地对待。这就是，权值空间中的先验分布被假设集中在原点附近。严格地讲，权值衰减并不是多层感知器复杂性正则化的正确形式，因为它并不符合式(4.95)的基本原理。然而，它是很简单的并且在一些应用中看起来工作得很好。

权值剔除 在这第二个复杂性正则化的过程中，复杂性惩罚定义为(Weigend et al., 1991)

$$\mathscr{E}_c(\mathbf{w}) = \sum_{i\in\mathscr{C}_{\text{total}}} \frac{(w_i/w_0)^2}{1+(w_i/w_0)^2} \tag{4.97}$$

其中 w_0 是预先指定的参数，w_i 是指网络中某个突触 i 的权值。集合$\mathscr{C}_{\text{total}}$是指网络中所有的突触连接。单独的惩罚项以对称的方式随 w_i/w_0 变化，如图4-22所示的那样。当$|w_i|\ll w_0$的时候，对于该权值的复杂性惩罚(代价)逼近于零。这个条件的含义是就所关注的从样本的

学习而言第 i 个突触权值是不可靠的从而应该从网络中剔除。而另一方面，当 $|w_i| \gg w_0$ 时，该权值的复杂性惩罚(代价)逼近最大值 1，这意味着 w_i 对反向传播学习过程是重要的。这样我们就看到式(4.97)中的惩罚项确实达到确认网络中有重要影响的突触权值这个期望目的。同时注意权值剔除过程包含权值衰减过程作为其特殊例子；特别地，对于大的 w_0，除了比例因子外式(4.97)简化为式(4.96)的形式。

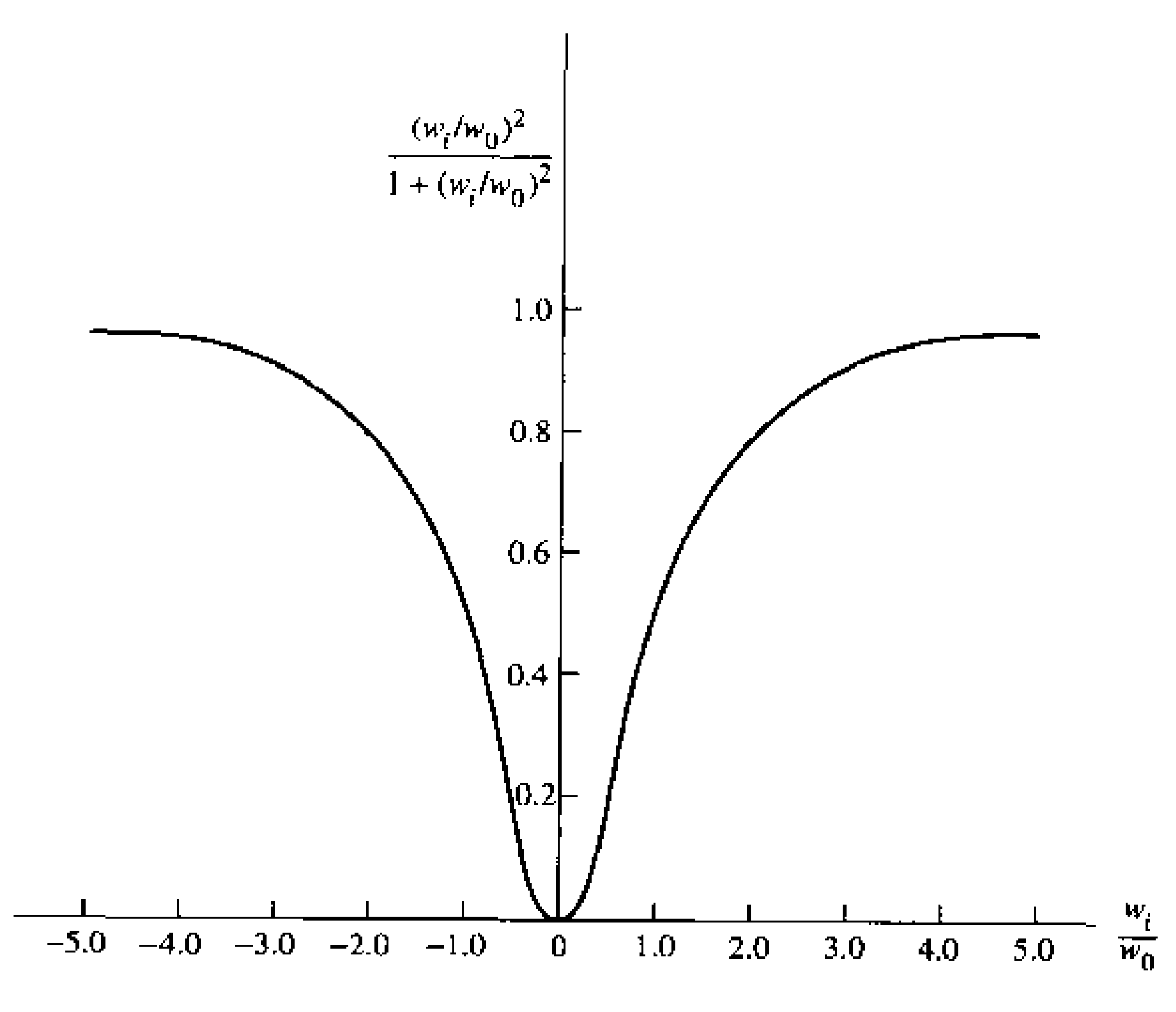


图 4-22 复杂性惩罚项 $(w_i/w_0)^2/[1+(w_i/w_0)^2]$ 关于 w_i/w_0 的图示

严格地讲，权值剔除过程同样不是多层感知器复杂性正则化的正确形式，因为它并不符合式(4.95)所指定的描述。虽然如此，在选择适当的参数 w_0 的情况下，它允许网络中的一些权值取得的值比利用权值衰减取得的值更大(Hush, 1997)。

220

逼近光滑器 在 Moody and Rögnvaldsson(1997)中，对于具有单个隐藏层和输出层上单个神经元的多层感知器，建议采用如下形式的复杂度惩罚项：

$$\mathscr{E}_c(\mathbf{w}) = \sum_{j=1}^{M} w_{oj}^2 \|\mathbf{w}_j\|^p \tag{4.98}$$

其中 w_{oj} 是输出层的权值，$\mathbf{w}_j$ 是隐藏层第 j 个神经元的权值向量；幂 p 定义为

$$p = \begin{cases} 2k-1 & \text{对于全局光滑器} \\ 2k & \text{对于局部光滑器} \end{cases} \tag{4.99}$$

其中 k 是 $F(\mathbf{x},\mathbf{w})$ 对 $\mathbf{x}$ 的微分的阶数。

对于一个多层感知器的复杂性正则化来说，逼近光滑器看来比权值衰减和权值剔除更精确。与早期的那些方法不同的是，它能完成如下两个工作：

1. 它区分隐藏层中和输出层中突触权值的作用。

221

2. 它捕获这两种权值集合之间的相互作用。

然而，它比权值衰减或者权值剔除具有更复杂的形式，因此在计算复杂度上有更多的要求。

基于 Hessian 矩阵的网络修剪

这第二个网络修剪方法的基本思想是利用误差曲面的二次导数信息得到网络复杂度和训练误差性能之间的折中方案。特别地，构造误差曲面的一个局部模型，*解析地预测突触权值的扰动所造成的影响*。构造这样一个模型结构的出发点是在运行点附近使用 Taylor 级数给出代价函

数$\mathscr{E}_{av}$的局部逼近，描述如下：

$$\mathscr{E}_{av}(\mathbf{w}+\Delta\mathbf{w}) = \mathscr{E}_{av}(\mathbf{w}) + \mathbf{g}^{T}(\mathbf{w})\Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^{T}\mathbf{H}\Delta\mathbf{w} + O(\|\Delta\mathbf{w}\|^{3}) \tag{4.100}$$

其中$\Delta\mathbf{w}$是运行点$\mathbf{w}$的扰动，$\mathbf{g}(\mathbf{w})$是在$\mathbf{w}$处的梯度向量。Hessian矩阵同样在$\mathbf{w}$点进行计算，因而，为了正确我们用$\mathbf{H}(\mathbf{w})$来表示它。在式(4.100)中并没有这么做仅仅是因为简化记号。

要求确认一组参数使得从多层感知器上删除它们而代价函数$\mathscr{E}_{av}$的值增长最小。为了用具体项解决这个问题，我们进行如下逼近：

1. *极值逼近*。我们假设参数仅在训练过程收敛(即网络被完全训练)之后才被从网络中删除。这个假设的含意就是参数的取值为误差曲面上一个局部最小或者全局最小。在这样一种情况下，梯度向量$\mathbf{g}$可以设为零因而可以忽略式(4.100)右边的$\mathbf{g}^{T}\Delta\mathbf{w}$项。否则显著性度量(将在后边定义)将对当前问题无效。

2. *二次逼近*。我们假设局部最小或者全局最小周围的误差曲面是近似"二次的"。因此同样可以忽略公式(4.100)中的更高次项。

在这两个假设之下，公式(4.100)被简单近似为

$$\Delta\mathscr{E}_{av} = \mathscr{E}(\mathbf{w}+\Delta\mathbf{w}) - \mathscr{E}(\mathbf{w}) \simeq \frac{1}{2}\Delta\mathbf{w}^{T}\mathbf{H}\Delta\mathbf{w} \tag{4.101}$$

最优脑损伤(Optimal Brain Damage, OBD)*过程*(LeCun et al., 1990b)通过更进一步的假设简化这个计算：假设Hessian矩阵$\mathbf{H}$是一个对角阵。然而，在*最优脑外科*(Optimal Brain Surgeon, OBS)*过程*(Hassibi et al., 1992)中并没有进行这样的假设；因此，它包含OBD过程作为它的一个特例。从这里开始，我们遵循OBS策略。 [222]

OBS的目标是置一个突触权值为零使得式(4.101)中给出的$\mathscr{E}_{av}$的递增增量最小化。令$w_i(n)$表示这个特别的突触权值。这个权值的删除等价于条件

$$\Delta w_i + w_i = 0$$

或者

$$\mathbf{1}_i^{T}\Delta\mathbf{w} + w_i = 0 \tag{4.102}$$

成立，其中$\mathbf{1}_i$是除了第i个元素等于单位1之外其他所有元素均为零的单位向量。我们现在可以重申OBS的目标如下(Hassibi et al., 1992)：

对权值向量增长变化$\Delta\mathbf{w}$最小化二次型$\frac{1}{2}\Delta\mathbf{w}^{T}\mathbf{H}\Delta\mathbf{w}$，使它满足约束条件$\mathbf{1}_i^{T}\Delta\mathbf{w}+w_i$为零，然后关于下标$i$求最小化。

这里进行两个层次上的最小化。一个最小化是当第i个权值向量置零后对仍保留的突触权值向量进行的；第二个最小化是对特定被修剪的向量进行的。

为了解决这个约束最优化问题，我们首先构建一个Lagrange算子

$$S = \frac{1}{2}\Delta\mathbf{w}^{T}\mathbf{H}\Delta\mathbf{w} - \lambda(\mathbf{1}_i^{T}\Delta\mathbf{w} + w_i) \tag{4.103}$$

其中λ是Lagrange乘子。然后求Lagrange函数S对$\Delta\mathbf{w}$的导数，应用式(4.102)的约束条件，并且利用矩阵的逆，我们发现权值向量$\mathbf{w}$中的最佳变化是

$$\Delta\mathbf{w} = -\frac{w_i}{[\mathbf{H}^{-1}]_{i,i}}\mathbf{H}^{-1}\mathbf{1}_i \tag{4.104}$$

Lagrange算子S对元素w_i的相应最优值是

$$S_i = \frac{w_i^2}{2[\mathbf{H}^{-1}]_{i,i}} \tag{4.105}$$

其中 $\mathbf{H}^{-1}$是 Hessian 矩阵 $\mathbf{H}$ 的逆，$[\mathbf{H}^{-1}]_{i,i}$是这个逆矩阵的第$(i,\ i)$个元素。假设第 i 个突触权值 w_i 被删除，对 $\Delta\mathbf{w}$ 进行优化而得到的 Lagrange 算子 S_i 称为 w_i 的显著性(saliency)。事实上，显著性 S_i 代表由于 w_i 的删除而导致的均方误差(性能标准)中的增长。注意显著性 S_i 是与 w_i^2 成正比的。这样小的权值在均方误差上具有小的影响。然而，从式(4.105)中我们看到显著性 S_i 同样是与逆 Hessian 矩阵的对角元素成反比的。这样如果$[\mathbf{H}^{-1}]_{i,i}$是小的，那么甚至小的权值也可能对均方误差有实质性的影响。

223 在 OBS 过程中，相应于最小特征值的权值被选为删除的权值。此外，剩余权值的最佳变化由公式(4.104)给出，这说明它们可以沿逆 Hessian 矩阵的第 i 列方向被校正。

Hassibi 等人在他们的论文中报告在一些基准的问题上 OBS 过程比其他通过使用权值衰减的过程产生更小的网络。同时报告 OBS 过程应用于包含单个隐藏层和 18 000 个权值的多层感知器 NETtalk 的结果，网络被修剪到仅有 1 560 个权值，这在网络的大小上有戏剧性的减少。归因于 Sejnowski and Rosenberg(1987)的 NETtalk 将在第 13 章中讲述。

计算 Hessian 矩阵的逆 Hessian 矩阵的逆 $\mathbf{H}^{-1}$是 OBS 过程的公式基础。当网络中自由参数 W 的数目很大的时候，计算 $\mathbf{H}^{-1}$的问题可能是难以处理的。设多层感知器被完全训练到误差曲面上的局部最小，下面我们描述一个计算 $\mathbf{H}^{-1}$的可控过程(Hassibi et al.,1992)。

为了简化表达，假设多层感知器具有单个输出神经元。然后对一个给定的训练集我们可以把代价函数表示为

$$\mathscr{E}_{av}(\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N}(d(n) - o(n))^2$$

其中 $o(n)$是第 n 个样本输入时网络的实际输出，$d(n)$是相应的期望响应，N 是训练集中样本的总数。输出 $o(n)$本身可以表示为

$$o(n) = F(\mathbf{w},\mathbf{x})$$

其中 F 是多层感知器实现的输入输出映射函数，$\mathbf{x}$ 是输入向量，$\mathbf{w}$ 是网络的突触权值向量。因此$\mathscr{E}_{av}$对 $\mathbf{w}$ 的一阶导数为

$$\frac{\partial \mathscr{E}_{av}}{\partial \mathbf{w}} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}(d(n) - o(n)) \tag{4.106}$$

$\mathscr{E}_{av}$对 $\mathbf{w}$ 的二阶导数或者 Hessian 矩阵是

$$\begin{aligned}\mathbf{H}(N) &= \frac{\partial^2 \mathscr{E}_{av}}{\partial \mathbf{w}^2}\\ &= \frac{1}{N}\sum_{n=1}^{N}\left\{\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)^T - \frac{\partial^2 F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}^2}(d(n) - o(n))\right\}\end{aligned} \tag{4.107}$$

在这里我们强调了 Hessian 矩阵对训练样本大小 N 的依赖性。

在网络是被完全训练的假设下，即代价函数$\mathscr{E}_{av}$被调整到误差曲面的一个局部最小值，说 $o(n)$近似于 $d(n)$是合理的。在这个条件下我们可以忽略第二项，这样公式(4.107)的逼近为

$$\mathbf{H}(N) \simeq \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)^{T} \tag{4.108}$$

224

为了简化符号，定义 $W\times 1$ 向量

$$\boldsymbol{\xi}(n) = \frac{1}{\sqrt{N}}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}} \tag{4.109}$$

它可以通过4.10节所述的过程来计算。然后我们就可以用递归的形式重写公式(4.108)如下：

$$\mathbf{H}(n) = \sum_{k=1}^{n}\boldsymbol{\xi}(k)\boldsymbol{\xi}^{T}(k) = \mathbf{H}(n-1)+\boldsymbol{\xi}(n)\boldsymbol{\xi}^{T}(n),\quad n=1,2,\cdots,N \tag{4.110}$$

这个递归正是所谓的矩阵逆引理应用的正确形式，它也称为 Woodbury 等式。

令 **A** 和 **B** 表示由关系

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}\mathbf{C}^{T}$$

定义的正定矩阵，其中 **C** 和 **D** 是另外两个矩阵。根据矩阵逆引理，矩阵 **A** 的逆定义为

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}\mathbf{C}(\mathbf{D}+\mathbf{C}^{T}\mathbf{B}\mathbf{C})^{-1}\mathbf{C}^{T}\mathbf{B}$$

对于式(4.110)中所述的问题我们有

$$\mathbf{A} = \mathbf{H}(\mathrm{n}),\mathbf{B}^{-1} = \mathbf{H}(n-1),\mathbf{C} = \boldsymbol{\xi}(n),\mathbf{D} = 1$$

因此应用矩阵逆引理得到对于 Hessian 矩阵求逆的递归计算公式：

$$\mathbf{H}^{-1}(n) = \mathbf{H}^{-1}(n-1) - \frac{\mathbf{H}^{-1}(n-1)\boldsymbol{\xi}(n)\boldsymbol{\xi}^{T}(n)\mathbf{H}^{-1}(n-1)}{1+\boldsymbol{\xi}^{T}(n)\mathbf{H}^{-1}(n-1)\boldsymbol{\xi}(n)} \tag{4.111}$$

注意式(4.111)中的分母是一个标量；因此直接计算它的倒数。这样，给定 Hessian 矩阵的逆过去的值 $\mathbf{H}^{-1}(n-1)$，我们就可以计算它由向量 $\boldsymbol{\xi}(n)$ 表示的第 n 个样本呈现后的更新值 $\mathbf{H}^{-1}(n)$。这个递归计算将继续到 N 个样本的整个集合被计算为止。为了初始化这个算法我们需要使 $\mathbf{H}^{-1}(0)$ 很大，因为根据式(4.111)它是持续地减少的。这个要求可以通过如下设定来满足：

$$\mathbf{H}^{-1}(0) = \delta^{-1}\mathbf{I} \tag{4.112}$$

225

其中 δ 是一个小的正数，**I** 是单位矩阵。这个初始化的形式保证 $\mathbf{H}^{-1}(n)$ 总是正定的。δ 的影响随着越来越多的样本出现在网络中而变得逐渐减少。

表4-6是脑外科算法的一个小结(Hassibi and Stork, 1992)。

表4-6 最优脑外科算法小结

1. 训练给定多层感知器至最小均方误差。
2. 利用4.10节所述过程计算向量

$$\boldsymbol{\xi}(n) = \frac{1}{\sqrt{N}}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}$$

   其中 $F(\mathbf{w},\mathbf{x}(n))$ 是由具有全部权值向量 $\mathbf{w}$ 的多层感知器实现的输入输出映射，$\mathbf{x}(n)$ 是输入向量。
3. 利用递归公式(4.111)计算 Hessian 矩阵的逆 $\mathbf{H}^{-1}$。
4. 寻找相应于最小显著性的 i：

$$S_i = \frac{w_i^2}{2[\mathbf{H}^{-1}]_{i,i}}$$

   其中 $[\mathbf{H}^{-1}]_{i,i}$ 是 $\mathbf{H}^{-1}$ 的第 (i,i) 个元素。如果显著性 S_i 远小于均方 $\mathscr{E}_{av}$，那么删除突触权值 w_i，并且执行第4步。否则，转第5步。
5. 通过应用如下调整校正网络中所有的突触权值：

$$\Delta\mathbf{w} = -\frac{w_i}{[\mathbf{H}^{-1}]_{i,i}}\mathbf{H}^{-1}\mathbf{1}_i$$

   转第2步。
6. 当不再有权值可以因为网络中均方误差没有大的增加而被删除的时候停止计算。(也许期望在该点重新训练网络。)

4.16 反向传播学习的优点和局限

反向传播算法作为指导多层感知器训练的最流行的算法而出现。基本上，它是一个梯度(导数)的技术而不是一个最优化技术。反向传播具有两个明显的性质：

- 局部计算简单。
- 它实现权值空间的随机梯度下降(对于突触权值更新按一个模型接一个模型的方式)。

多层感知器背景下的反向传播学习的这两个属性导致它的优点和缺点。

连接机制

反向传播算法是依靠局部计算来发现神经网络信息处理能力的一个*连接论者范例*的例子。计算限制的这种形式称为*局部约束*，它是指单个神经元实现的计算惟一受那些与它有物理接触的神经元的影响。在人工神经网络的设计中提倡利用局部计算有三个主要的理由：

[226] 1. 实现局部计算的人工神经网络常常支持生物神经网络的类比。

2. 局部计算的使用允许极大地减弱由于硬件错误所导致的性能下降，因此为容错网络设计提供基础。

3. 局部计算支持使用作为人工神经网络实现的有效方法的并行体系结构。

按相反的顺序来讨论这三点，第三点在反向传播学习中被完全验证。特别地，反向传播算法已经被许多研究者在并行计算机上成功地实现了，并且已经开发用硬件实现多层感知器的VLSI体系结构(Hammerstrom,1992a,1992b)。正如在Kerlirzin and Vallet(1993)的研究中所述的那样，第二点的验证可由反向传播算法的应用中采取某些防范措施而得到。对于第一点，和反向传播学习的生物似真性有关，基于如下理由它受到严重的质疑(Shepherd,1990b;Crick,1989;Stork,1989)：

1. 在一个多层感知器神经元之间的双向突触连接可以假设权值是兴奋的或者是抑制的。然而，在真实的神经网络系统中，神经元经常表现为一个或者另一个。这就是在神经网络模型中所作的不真实的假设中最严重的一个。

2. 在一个多层感知器中，忽略了荷尔蒙的和其他类型的全局通信的类型。在真实的神经元系统中，这些全局通信对于例如激励、注意和学习的状态设置功能是关键的。

3. 在反向传播学习中，一个突触权值是通过一个前突触活动和一个独立于后突触活动的误差(学习)信号来修改的。从神经生物学证据表明是另一种情况。

4. 从神经生物学的角度来看，反向传播学习的实现要求信息沿着轴突迅速地反向传播。在脑中实际发生的这样操作看起来简直是不可能的。

5. 反向传播学习意味着一个"教师"的存在，这在脑中将假设存在一个具有特殊性质的神经元集合。这样的神经元的存在在生物学上是难以置信的。

然而，这些神经生物学上的疑虑并没有减少反向传播学习作为信息处理的一个工具在工程上的重要性，这通过它在无数大不相同的领域中的成功应用得到了证明，其中包括神经生物现象的仿真在内(例如，见Robinson(1992))。

特征检测

正如4.9节所讨论的那样，通过反向传播算法训练的多层感知器的隐藏神经元作为特征

检测器扮演着重要的角色。利用多层感知器的这个重要性质的一个新方法是使用它作为复制器或者恒等映射(Rumelhart et al.,1986b;Cottrel et al.,1987)。图4-23表明对于使用单个隐藏层的多层感知器情况下这是如何完成的。网络构形满足如下的结构要求，正如图4-23a表明的那样： [227]

- 输入和输出层神经元数目具有相同的大小 m。
- 隐藏层的神经元个数 M 小于 m。
- 网络是完全连接的。

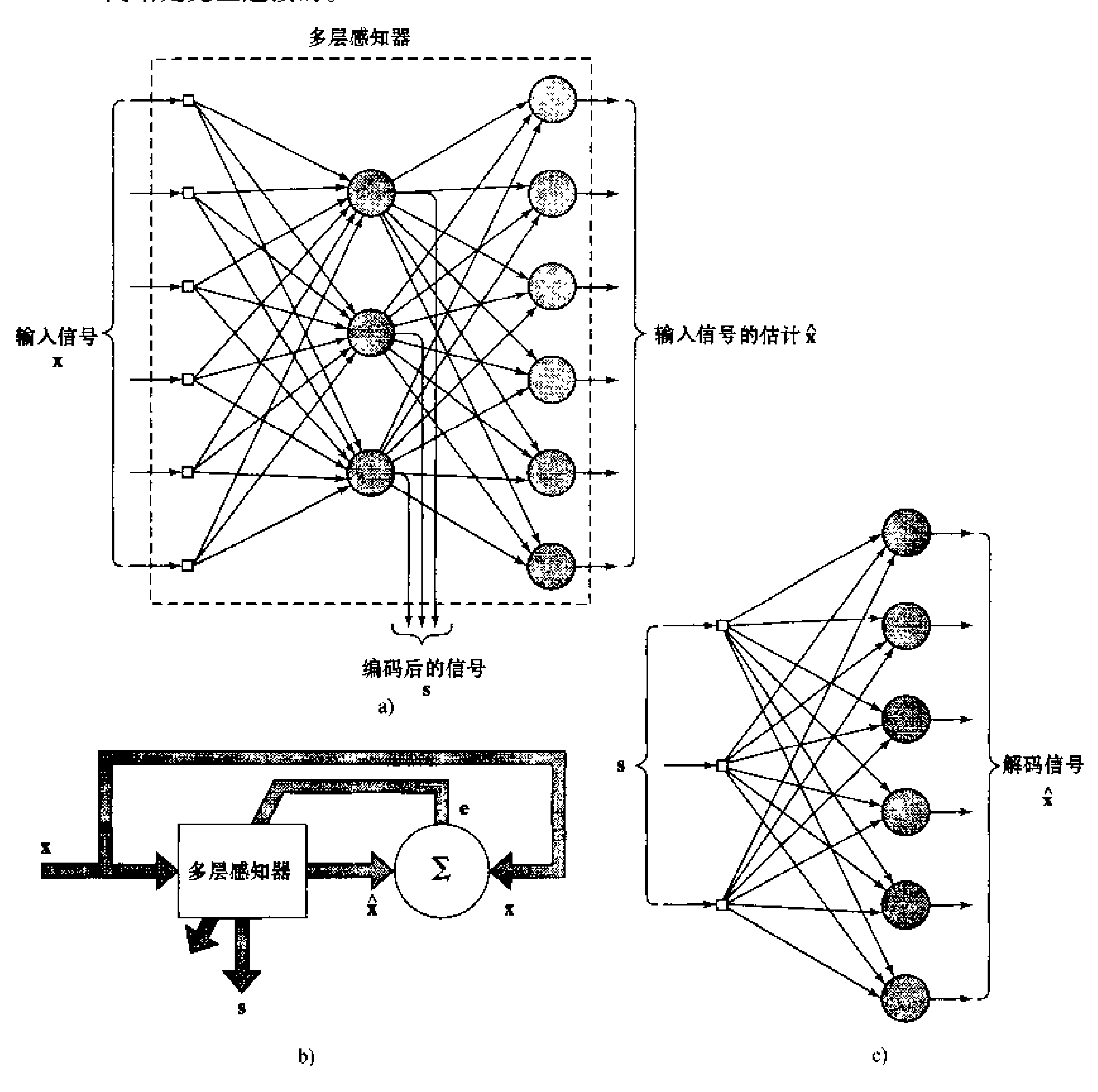


图4-23
a)具有一个隐藏层的作为编码器的复制器网络(恒等映射) b)复制网络监督训练的方框图 c)作为解码器的复制器网络部分

一个给定的模式 **x** 同时作为输入层的刺激和输出层的期望响应。输出层的实际响应 $\hat{\mathbf{x}}$ 是打算用作 **x** 的“估计”。通过常用的方法使用反向传播算法训练网络，估计误差向量($\mathbf{x}-\hat{\mathbf{x}}$)作为误差信号处理，如图4-23b所示。这个训练是在无监督情形下完成的(即不需要教师)。借 [228]

助多层感知器的设计所建立的特殊结构优点，通过它的隐藏层约束网络以实现恒等映射。输入模式的一个编码形式，用 s 表示，它是在隐藏层的输出中产生的，如图 4-23a 所示。事实上，完全训练的多层感知器充当着“编码器”的角色。为了重构初始输入模式 $\mathbf{x}$ 的估计 $\hat{\mathbf{x}}$(即实现解码)，我们将编码信号应用于复制器网络隐藏层，如图 4-23c 所示。事实上，后面的网络扮演“解码器”的角色。如果我们使得隐藏层的大小 M 与输入/输出层大小 m 相比越小，那么图 4-23a 的结构作为一个数据压缩系统的作用就越大[12]。

函数逼近

通过反向传播算法训练的多层感知器自身表明是一个嵌套 sigmoid 函数，在单个输出的情形下用紧凑形式写为

$$F(\mathbf{x},\mathbf{w}) = \varphi\Big(\sum_k w_{ok}\varphi\Big(\sum_j w_{kj}\varphi\Big(\cdots\varphi\Big(\sum_i w_{li}x_i\Big)\Big)\Big)\Big) \tag{4.113}$$

其中 $\varphi(\cdot)$是常用 sigmoid 激活函数，w_{ok}是从最后一个隐藏层的神经元 k 到单个输出神经元 o 的突触权值，依此类推得到其他突触权值，x_i 是输入向量 $\mathbf{x}$ 的第 i 个元素。权值向量 $\mathbf{w}$ 表示突触权值的完整集合，其排列顺序首先按层，然后按每层中的神经元，最后按神经元中的突触。式(4.113)中嵌入非线性函数的设计在经典逼近论中是不常见的。正如 4.13 节讨论的它是一个通用逼近器。

在逼近背景下，使用反向传播学习提供另一个有用的性质。直觉的知识暗示具有光滑激活函数的多层感知器的输出函数的导数应该同样逼近未知输入－输出映射的导数。在 Hornik et al.(1990)中介绍了这个结果的证明。实际上，证明多层感知器能逼近传统意义下不可微的函数，但拥有像在分段可微函数情形下的广义导数的函数。Hornik 等人报告的逼近结果提供了以前利用多层感知器逼近一个函数和它的导数所缺少的理论根据。

计算的效率

算法的计算复杂度通常是用乘法、加法的次数和它的实现所涉及的存储量来衡量的，如第 2 章所讨论的那样。一个学习算法从一次迭代到下一次迭代，若它计算复杂度更新的可调整参数的数目是多项式的，我们就说这个算法是计算有效的。在这个基础上，它也可以说是反向传播算法是计算有效的。特别地，在使用它进行包含全部的突触权值 W(包括偏置)的多层感知器的训练中，它的计算复杂度在 W 中是线性的。反向传播算法的这个重要性质可以通过检查如 4.5 节所述的完成前向通过和反向通过所涉及的计算而容易得到证明。在前向通过中，计算涉及的突触权值是那些网络中不同神经元的诱导局部域所属的权值。这里我们从式(4.44)看到这些计算对网络的突触权值是线性的。在反向通过中，涉及突触权值的仅有的计算是那些分别由式(4.46)和(4.47)所述的属于(1)隐藏神经元的局部梯度，和(2)突触权值自身的更新。在这里我们同样可以看到这些计算对网络的突触权值全部是线性的。因此得出结论，反向传播算法的计算复杂度对 W 是线性的，即它是 $O(W)$。

229

灵敏度分析

从使用反向传播学习中得到的另一个计算上的好处是它提供一个有效的方法，通过它我们可以进行由这个算法实现的输入输出映射的灵敏度分析。输入输出映射函数 F 关于一个

参数的灵敏度，以 ω 表示，定义为

$$S_{\omega}^{F} = \frac{\partial F/F}{\partial \omega/\omega} \tag{4.114}$$

然后考虑一个经过反向传播算法训练的多层感知器。令函数 $F(\mathbf{w})$ 为网络实现的输入输出映射；$\mathbf{w}$ 表示网络中包含的所有突触权值(包括偏置)向量。在 4.10 节中我们证明了函数 $F(\mathbf{w})$ 对权值向量 $\mathbf{w}$ 中所有元素的偏导数是可以进行有效计算的。具体地，检查式(4.81)、(4.83)和式(4.114)，我们知道这些偏导数计算涉及的复杂性对网络包含权值的总数 W 是线性的。这种线性关系与问题的突触权值在计算链中出现的位置无关。

鲁棒性

在第 3 章中我们指出，LMS 算法中能量小的扰动只会引起小的估计误差，从这个角度来看它是鲁棒的。如果固有的观察模型是线性的，LMS 算法是一个 H^{∞} 最优滤波器(Hassibi et al.,1993,1996)。这意味着 LMS 算法最小化由估计误差的扰动带来的最大能量增益。

从另一方面来看，如果固有的观察模型是非线性的，Hassibi 和 Kailath(1995)证明反向传播算法是局部 H^{∞} 最优滤波器。这里使用的“局部”术语是指反向传播算法中使用的权值向量初始值充分靠近权值向量的最优值 $\mathbf{w}^{*}$ 以确保该算法不陷入一个坏的局部最小中。用概念性的说法，看到 LMS 和反向传播算法属于同一类型的 H^{∞} 最优滤波器是令人满意的。

收敛性

230

反向传播算法在权值空间中对于误差曲面上的梯度使用“瞬时估计”。因此该算法在本质上是随机的；也就是说，它在误差曲面上具有通过在真实方向附近的锯齿形路线趋于最小点的倾向。其实，反向传播学习是最初由 Robbins 和 Monro(1951)提出的所谓随机逼近的统计学方法的一个应用。因此，它倾向于缓慢收敛。我们可以验明这个性质的两个基本原因(Jacobs,1988)：

1. 误差曲面沿着一个权值方向是相当平坦的，这意味着误差曲面对这个权值的导数在数量上是很小的。在这样的情况下，应用于这个权值的调整是很小的，因此在网络误差性能上产生重大的降低可能要求这个算法的多次迭代。或另一方面，误差曲面沿着一个权值方向是高度弯曲的，在这种情形下误差曲面对该权值的导数在数量上是很大的。在这第二种情况下，应用于该权值的调整是很大的，这可能会导致该算法越过误差曲面的最小点。

2. 负梯度向量的方向(即代价函数对权值向量的负导数)可能指向远离误差曲面的最小值：因此应用于权值的调整可能导致算法往错误的方向进行。

因此，反向传播学习的收敛速度倾向于相当缓慢，这可能使得计算起来非常困难。根据 Saarinen et al.(1992)的实验研究，反向传播算法的局部收敛速度是线性的，这通过 Jacobi 矩阵和 Hessian 矩阵几乎是秩亏损的而得到证明。这些都是神经网络训练问题固有的病态性的结果。Saarinen 等人用两种方法之一解释反向传播学习的线性局部收敛速度：

- 较高阶的方法要求更多的计算量未必收敛得更快，在这个意义上反向传播(梯度下降)是可接受的；
- 大规模神经网络的训练问题的实施有如此大的固有困难以至于没有任何监督学习的策略是可行的，而使用如像预处理的其他方法可能是必需的。

在4.17节中我们更全面地探讨收敛问题，并且在第8章中探讨输入的预处理问题。

231

局部最小值

对反向传播算法性能造成影响的误差曲面的另一个特点是除了全局最小值之外的局部最小值(即孤立凹槽)的出现。由于反向传播学习基本上是一个爬山技术，因此它存在陷入局部最小值的危险，此处突触权值的每个微小变化都引起代价函数的增长。但在权值空间的别的某个地方存在另外一个突触权值的集合，它的代价函数的值比在网络被停止处的局部最小值更小。很明显不希望使学习进程在局部最小值处停止，特别是如果它是处于离全局最小值很远的话。

反向传播学习中局部最小值的问题在 Minsky and Papert(1988)经典著作的扩充版本的结语中被提了出来，结语的绝大部分注意力都集中讨论分为两册的 Rumelhart 和 McClelland (1986)著作：《*Parallel Distributed Processing*》。在这本书的第8章中声称对于反向传播学习来说，陷入一个局部最小值在一个实际问题中是罕见的。Minsky 和 Papert 通过指出模式识别整个历史过程的相反表现进行反驳。Gori 和 Tesi(1992)描述一个简单的例子，尽管模式中一个非线性的可分集合能够通过选择具有单个隐藏层的网络进行学习，但是反向传播学习还是可能在一个局部最小值处停止[13]。

规模

在原则上，诸如由反向传播算法训练的多层感知器之类的神经网络提供通用计算机器的潜在可能。然而，要充分实现这种潜能，我们必须克服规模(scaling)问题，它是指随计算任务在大小和复杂性上的增加网络表现的优劣(如由训练所需时间和可得到的最优泛化性能来衡量)的问题。在度量计算任务大小和复杂度的许多可能的办法中，由 Minsky 和 Papert (1969, 1988)定义的谓词阶(predicate order)提供了最有用和最重要的标准。

为了解释一个谓词意味着什么，令 $\Psi(X)$表示一个只能有两个取值的函数。通常我们取 $\Psi(X)$的两个值为0和1。但通过取值为假(FALSE)或真(TRUE)，可以认为 $\Psi(X)$是一个谓词，即一个可变的陈述，其真和假依赖于变量 X 的选择。例如，我们可以写出

$$\Psi_{\text{CIRCLE}}(X) = \begin{cases} 1 & \text{若图形 } X \text{ 是一个圆} \\ 0 & \text{若图形 } X \text{ 不是一个圆} \end{cases} \tag{4.115}$$

使用谓词的思想，Tesauro and Janssens(1988)实现了一个涉及使用由反向传播算法训练的多层感知器来学习计算奇偶函数的实验研究。奇偶函数是定义如下的布尔谓词：

$$\Psi_{\text{PARITY}}(X) = \begin{cases} 1 & \text{若 } |X| \text{ 是奇数} \\ 0 & \text{否则} \end{cases} \tag{4.116}$$

它的阶数等于输入的个数。Tesauro and Janssens 进行的这个实验显示，网络学习计算奇偶函数所需的时间与输入个数(即计算的谓词阶数)呈指数关系，并且使用反向传播算法学习任意复杂的函数的计划可能是过分乐观的。

232

一般认为对一个多层感知器进行完全连接是失策的。因此，在此背景下，我们可以提出如下问题：给定一个不应被完全连接的多层感知器，网络的突触连接将如何分配？这个问题在小规模的应用情况并不是主要考虑的问题，但它对利用反向传播学习解决现实世界中大规模的问题的成功应用是至关重要的。

减轻规模问题的一个有效办法是发展对当前问题的认识(可能是通过神经生物学的类比)并利用它增加多层感知器体系结构设计的灵活性。特别地，网络体系结构和加于网络突触权值上的约束应该这样设计使得关于任务的先验知识合并到网络的组成中去。这种设计策略在4.19节中在关于光学字符识别的问题中说明。

4.17 反向传播学习的加速收敛

在前一节中阐明了反向传播算法收敛速率可能缓慢的主要原因。本节我们讨论一些得到的启发，它们为思考如何通过学习率的调整以加速反向传播学习的收敛提供有用的方针。具体的启发如下(Jacobs,1988)：

启发 1 代价函数的每一个可调整网络参数都应具有自己的学习率参数。

在这里我们注意反向传播算法可能缓慢地收敛是因为使用固定的学习率参数不能适合于误差曲面地每一部分。换句话说，一个突触权值调节的适宜的学习率参数是不必适宜于网络中其他突触权值的调节的。启发1通过为网络中每个可调节的突触权值(参数)指定不同的学习率参数认知这个事实。

启发 2 每一个学习率参数都应该被允许在每次迭代中取不同的值。

沿着单个权值维的不同区域，误差曲面通常有不同的行为。为了适应这种变化，启发2规定学习参数在每次迭代中不同。有趣的是，这个启发在线性单元的情形中被明确地建立(Luo,1991)。

启发 3 当代价函数对一个突触权值的导数在算法中几次连续迭代具有相同的代数符号的时候，这个特殊权值的学习率参数应该被增加。

233

在权值空间中当前运行点所处误差曲面沿一个特别的权值维可能是相当平坦的部分。这可以导致代价函数关于权值的导数(即误差曲面的梯度)在连续几次算法迭代中保持相同代数符号，因此指向相同的方向。启发3规定在这样的情形下可以通过适当增加学习率参数来减少通过误差曲面的平坦部分所需的迭代次数。

启发 4 当代价函数对个别突触权值的导数的代数符号对于连续几次算法迭代发生改变的时候，该权值的学习率参数应该减少。

当在权值空间中当前运行的点所位于误差曲面的部分沿所讨论的权值维呈现峰值和深谷(即曲面高度弯曲)的时候，代价函数对该权值的导数在这次迭代到下次迭代时改变它的符号是可能的。为了防止权值调节出现振荡，启发4规定该特殊权值的学习率参数应该适当地减少。

值得注意的是，根据这些启发对每个突触权值使用不同的和随时间变化的学习率参数，从基本上改变了反向传播算法。特别地，被修改后的算法不再进行最陡下降方向的搜索。更准确地说，应用于突触权值的调整是基于(1)误差曲面对权值的偏导数，和(2)在权值空间当前运行点上误差曲面在沿不同权值维的曲率估计。

此外，所有4个启发都满足局部约束，这是反向传播学习的固有特征。不幸的是，对局部约束的坚持限制了这些启发的领域，因为存在它们不能工作的误差曲面。然而，根据这些启发对反向传播算法的修改确实具有实用价值[14]。

4.18 作为最优化问题看待的有监督学习

在本节用一种与前面几节讨论有很大不同的关于有监督学习的观点。特别地，我们把多

层感知器的监督训练看作是一个数值最优化问题. 在这个背景下我们首先指出使用有监督学习的多层感知器的误差曲面是突触权值向量 $\mathbf{w}$ 的高度非线性函数。令$\mathscr{E}_{av}(\mathbf{w})$表示在训练样本上平均的代价函数。使用 Taylor 级数在误差曲面当前点 $\mathbf{w}(n)$附近我们可以展开$\mathscr{E}_{av}(\mathbf{w})$，例如，如式(4.100)所描述的，这里重写为依赖于 n 的形式：

$$\mathscr{E}_{av}(\mathbf{w}(n)+\Delta\mathbf{w}(n)) = \mathscr{E}_{av}(\mathbf{w}(n)) + \mathbf{g}^T(n)\Delta\mathbf{w}(n) + \frac{1}{2}\Delta\mathbf{w}^T(n)\mathbf{H}(n)\Delta\mathbf{w}(n) + (\text{三次和更高次项}) \tag{4.117}$$

其中 $\mathbf{g}(n)$是局部梯度向量，定义为

234

$$\mathbf{g}(n) = \left.\frac{\partial \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}(n)} \tag{4.118}$$

$\mathbf{H}(n)$是局部 Hessian 矩阵，定义为

$$\mathbf{H}(n) = \left.\frac{\partial^2 \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}^2}\right|_{\mathbf{w}=\mathbf{w}(n)} \tag{4.119}$$

使用总体平均代价函数$\mathscr{E}_{av}(\mathbf{w})$假设为集中式学习。

在以反向传播算法为例的最陡下降方向方法中，应用于突触权值向量 $\mathbf{w}(n)$的调节量 $\Delta\mathbf{w}(n)$定义为

$$\Delta\mathbf{w}(n) = -\eta\mathbf{g}(n) \tag{4.120}$$

其中 η 为学习率参数。事实上，最陡下降方向方法是在运行点 $\mathbf{w}(n)$局部邻域对代价函数的线性逼近基础上进行计算的。在这样的处理中，它依赖梯度向量 $\mathbf{g}(n)$作为关于误差曲面局部信息的惟一来源。这个限制具有一个有利的效果：实现的简单性。不幸的是，它同样具有一个不利的影响：缓慢的收敛速度，特别是在大规模问题的情形下这是令人烦恼的。在权值更新的公式中包含动量项是使用误差曲面二阶信息的大胆尝试，这是具有某些帮助的。然而，由于在必须由设计者"调整"的参数列表中增加一项，它的使用使得训练过程的管理更费时间。

为了使多层感知器的收敛性能有显著的改善(与反向传播学习相比)，必须使用训练过程的高阶信息。我们可以通过调用误差曲面在当前点 $\mathbf{w}(n)$周围的二次逼近来实现。然后从式(4.117)可以发现应用于突触权值向量 $\mathbf{w}(n)$的调整量的最优值 $\Delta\mathbf{w}(n)$由下式给出：

$$\Delta\mathbf{w}^*(n) = \mathbf{H}^{-1}(n)\mathbf{g}(n) \tag{4.121}$$

其中 $\mathbf{H}^{-1}(n)$是 Hessian 矩阵 $\mathbf{H}(n)$的逆，假设它是存在的。式(4.121)是 Newton 方法的核心。如果代价函数$\mathscr{E}_{av}(\mathbf{w})$是二次的(即式(4.117)中的三次和更高次项为零)，那么 Newton 方法一次迭代后收敛到最优值位置。然而，Newton 方法对多层感知器的有监督训练的实际应用受到如下因素的阻碍：

- 它要求计算 Hessian 矩阵的逆 $\mathbf{H}^{-1}(n)$，这可能在计算上是昂贵的。
- 为了使 $\mathbf{H}^{-1}(n)$是可计算的，$\mathbf{H}(n)$必须是非奇异的。在 $\mathbf{H}(n)$为正定的情况下，当前点 $\mathbf{w}(n)$周围的误差曲面可以描述为"凸碗状"。不幸的是，并不能保证多层感知器误差曲面的 Hessian 矩阵总是符合这样的描述。而且，还有 Hessian 矩阵秩亏损的潜在问题(即并不是所有的 $\mathbf{H}$ 的列都线性无关)，这是由于网络训练问题中固有的病态性所造成的(Saarinen et al., 1992)；这只会使得计算任务更加困难。

235

- 当代价函数$\mathscr{E}_{av}(\mathbf{w})$是非二次的时候，Newton 方法的收敛性得不到保证，这使得它不适合于训练多层感知器。

为了克服其中某些困难，我们可以使用拟 Newton 方法，它仅仅要求梯度向量 $\mathbf{g}$ 的一个估计值。这种 Newton 方法的修正不经过计算矩阵的逆而直接得到逆矩阵 $\mathbf{H}^{-1}$ 保持正定的估计。通过使用这样的估计，拟 Newton 方法保证在误差曲面上是下降的。然而，我们仍然有一个 $O(W^2)$ 的计算复杂性，其中 W 是权值向量 $\mathbf{w}$ 的大小。因此拟 Newton 方法在计算上是不切实际的，除非对一个非常小规模的神经网络进行训练。关于拟 Newton 方法的讨论将在本节后面给出。

另一类型的二阶最优化方法包括共轭梯度方法，它被认为是一种介于最陡梯度方法和 Newton 方法之间的方法。使用共轭梯度方法的动机是期望加速在最陡梯度方法中经历的特别缓慢的收敛速度，同时避免在 Newton 方法中要求对 Hessian 矩阵的估值、存储和求逆。在二次最优化方法中，广为人知的是共轭梯度方法也许是可用于大规模问题的惟一方法，大规模问题就是具有几百个或几千个可调整参数的问题(Fletcher, 1987)。因此它非常适合于训练多层感知器，典型的应用包括函数逼近、控制和时间序列分析(即回归分析)。

共扼梯度方法

共扼梯度方法属于人所共知的共扼方向方法的二阶最优化方法的一类。我们通过考虑二次函数

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x} + c \tag{4.122}$$

的最小化来开始这些方法的讨论，其中 $\mathbf{x}$ 是一个 $W \times 1$ 参数向量，$\mathbf{A}$ 是 $W \times W$ 对称正定矩阵，$\mathbf{b}$ 是 $W \times 1$ 向量，c 是标量。二次函数 $f(\mathbf{x})$ 的最小化是通过赋予 $\mathbf{x}$ 如下惟一值得到的：

$$\mathbf{x}^* = \mathbf{A}^{-1}\mathbf{b} \tag{4.123}$$

这样 $f(\mathbf{x})$ 的最小化和求解方程 $\mathbf{A}\mathbf{x}^* = \mathbf{b}$ 的线性系统就是等价问题。

给定矩阵 $\mathbf{A}$，如果下述条件满足，我们称非零向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(W-1)$ 的集合是 $\mathbf{A}$－共扼的(即在矩阵 $\mathbf{A}$ 下互不干扰)：

$$\mathbf{s}^T(n)\mathbf{A}\mathbf{s}(j) = 0 \qquad 所有\ n \neq j \tag{4.124}$$

236

如果 $\mathbf{A}$ 等于单位矩阵，共轭就等同于通常的正交性概念。

例 4.1 为了解释 $\mathbf{A}$－共轭向量，考虑图 4-24a 所示属于二维问题的情形。图中所示椭圆轨迹对应于方程(4.122)在

$$\mathbf{x} = [x_0, x_1]^T$$

对二次函数 $f(\mathbf{x})$ 指定某个常数值的图形，图 4-24a 也包括一对关于矩阵 $\mathbf{A}$ 共轭的方向向量。假定我们通过变换

$$\mathbf{v} = \mathbf{A}^{1/2}\mathbf{x}$$

定义一个新的与 $\mathbf{x}$ 相关的参数向量 $\mathbf{v}$，其中 $\mathbf{A}^{1/2}$ 是 $\mathbf{A}$ 的平方根。这样图 4-24a 中椭圆轨迹就被变换为图 4-24b 所示的圆形轨迹，图 4-24a 中 $\mathbf{A}$－共轭的方向向量对也被

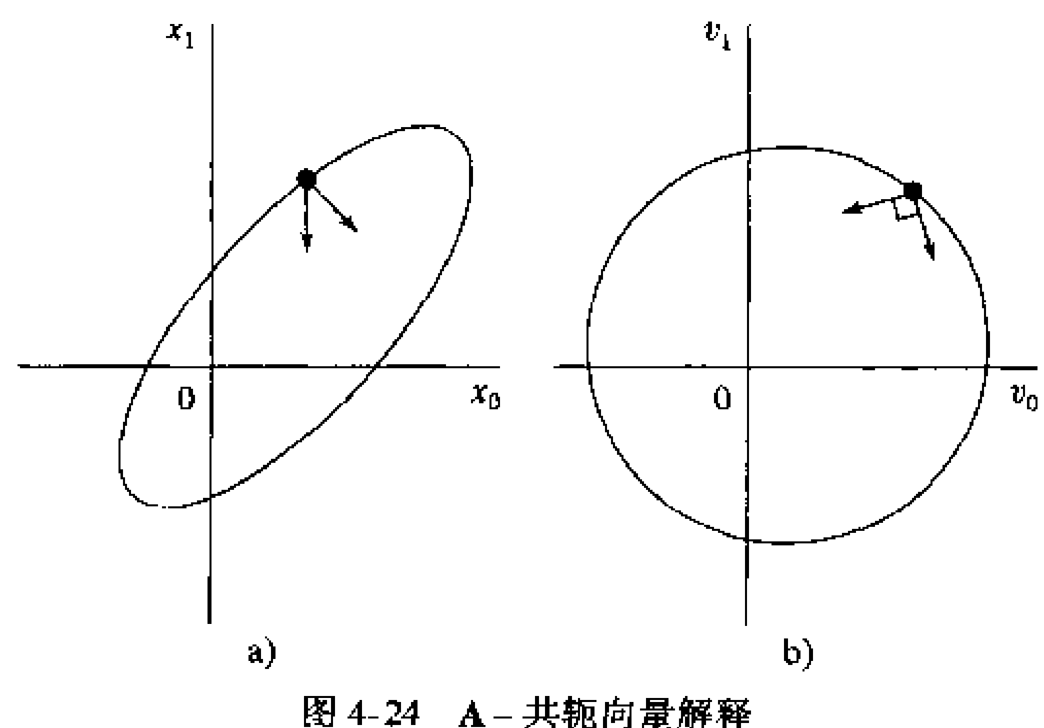


图 4-24 A－共轭向量解释
a)二维权值空间的椭圆轨迹 b)椭圆轨迹到圆形轨迹的变换

转换为图4-24b中的一对正交方向向量。 ■

关于 **A** - 共轭向量的一个重要性质是它们是线性无关的。我们可以用反证法证明这个性质。令这些向量的其中之一，比如 $\mathbf{s}(0)$，用其余 $W-1$ 个向量的线性组合表示如下：

$$\mathbf{s}(0) = \sum_{j=1}^{W-1} \alpha_j \mathbf{s}(j)$$

两边乘以 **A** 并用 $\mathbf{s}(0)$ 和 $\mathbf{As}(0)$ 作内积得到

$$\mathbf{s}^T(0)\mathbf{As}(0) = \sum_{j=1}^{W-1} \alpha_j \mathbf{s}^T(0)\mathbf{As}(j) = 0$$

然而，有两个原因使得二次型 $\mathbf{s}^T(0)\mathbf{As}(0)$ 不可能为零：矩阵 **A** 是被假设为正定的，向量 $\mathbf{s}(0)$ 定义为非零。因此可以得出 **A** - 共轭的向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(W-1)$ 不能是线性相关的；也就是，它们必须是线性无关的。

[237]

对于给定一个 **A** - 共轭的向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(W-1)$ 的集合，相应的二次误差函数 $f(\mathbf{x})$ 的无约束最优化*共轭方向方法*定义为(Luenberger, 1973; Fletcher, 1987; Bertsekas, 1995)

$$\mathbf{x}(n+1) = \mathbf{x}(n) + \eta(n)\mathbf{s}(n), \qquad n = 0,1,\cdots,W-1 \tag{4.125}$$

其中 $\mathbf{x}(0)$ 是任意的开始向量，$\eta(n)$ 是由

$$f(\mathbf{x}(n) + \eta(n)\mathbf{s}(n)) = \min_{\eta} f(\mathbf{x}(n) + \eta\mathbf{s}(n)) \tag{4.126}$$

定义的标量。通过选择 η 对某个固定的 n 寻找使函数 $f(\mathbf{x}(n) + \eta\mathbf{s}(n))$ 最小化的过程称为线搜索，这表示一维最小化问题。

根据式(4.124)，(4.125)和(4.126)，我们提供如下观察结果：

1. 由于 **A** - 共轭的向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(W-1)$ 线性无关，它们组成 $\mathbf{w}$ 的向量空间的一组基。

2. 更新公式(4.125)和式(4.126)的线最小化导出学习率参数相同的公式，即

$$\eta(n) = -\frac{\mathbf{s}^T(n)\mathbf{A}\boldsymbol{\varepsilon}(n)}{\mathbf{s}^T(n)\mathbf{As}(n)}, \qquad n = 0,1,\cdots,W-1 \tag{4.127}$$

其中 $\boldsymbol{\varepsilon}(n)$ 是*误差向量*，定义为

$$\boldsymbol{\varepsilon}(n) = \mathbf{x}(n) - \mathbf{x}^* \tag{4.128}$$

3. 从任意一个点 $\mathbf{x}(0)$ 出发，共轭方向方法确保能在最多迭代 W 次中找到二次函数 $f(\mathbf{x})$ 的最优解 $\mathbf{x}^*$。

共轭方向方法的主要性质描述为(Luenberger, 1984; Fletcher, 1987; Bertsekas, 1995)：

在连续的迭代中，共轭方向方法在逐渐扩张的线性向量空间上最小化二次函数 $f(\mathbf{x})$，最终包含 $f(\mathbf{x})$ 的全局最小值。

特别地，对于每次迭代 n，迭代结果 $\mathbf{x}(n+1)$ 在通过某个任意点 $\mathbf{x}(0)$ 并且由 **A** - 共轭的向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(n)$ 扩展成的线性向量空间 $\mathscr{D}_n$ 上使函数 $f(\mathbf{x})$ 最小化，表示为

$$\mathbf{x}(n+1) = \arg\min_{\mathbf{x}\in\mathscr{D}_n} f(\mathbf{x}) \tag{4.129}$$

其中空间 $\mathscr{D}_n$ 定义为

$$\mathscr{D}_n = \left\{ \mathbf{x}(n) \mid \mathbf{x}(n) = \mathbf{x}(0) + \sum_{j=0}^{n} \eta(j)\mathbf{s}(j) \right\} \tag{4.130}$$

[238]

为了使共轭方向方法起作用，我们要求具备一个 **A** - 共轭的向量 $\mathbf{s}(0)$，$\mathbf{s}(1)$，…，

$\mathbf{s}(W-1)$的集合可用。在这种方法的一种称为共轭梯度方法[15]的特殊形式中，随着这个方法逐步进行二次函数$f(\mathbf{x})$相继的梯度向量的$\mathbf{A}$－共轭形式产生相继的方向向量，因此以此来命名这种方法。这样，除了$n=0$之外，方向向量的集合$\{\mathbf{s}(n)\}$并不是预先指定的，相反它是在该方法的相继的步骤中串行决定的。

定义残差作为最陡下降方向：

$$\mathbf{r}(n) = \mathbf{b} - \mathbf{A}\mathbf{x}(n) \tag{4.131}$$

进而通过$\mathbf{r}(n)$和$\mathbf{s}(n-1)$的线性组合，表示为

$$\mathbf{s}(n) = \mathbf{r}(n) + \beta(n)\mathbf{s}(n-1), \qquad n = 1,2,\cdots,W-1 \tag{4.132}$$

其中$\beta(n)$是需要确定的一个比例因子。利用方向向量$\mathbf{A}$－共轭的性质，方程的两边乘以$\mathbf{A}$，并将结果表达式和$\mathbf{s}(n-1)$作内积，然后求解$\beta(n)$的结果表达式，我们得到

$$\beta(n) = -\frac{\mathbf{s}^T(n-1)\mathbf{A}\mathbf{r}(n)}{\mathbf{s}^T(n-1)\mathbf{A}\mathbf{s}(n-1)} \tag{4.133}$$

通过式(4.132)和(4.133)，我们发现这样得到的向量$\mathbf{s}(0)$，$\mathbf{s}(1)$，…，$\mathbf{s}(W-1)$确实是$\mathbf{A}$－共轭的。

根据递归公式(4.132)产生方向向量依赖于系数$\beta(n)$。由于$\beta(n)$目前的表示形式，对$\beta(n)$的计算公式(4.133)要求矩阵$\mathbf{A}$的知识。出于计算上的原因，希望不利用$\mathbf{A}$的明显知识的情况下对$\beta(n)$进行计算。这样的计算可以通过两个不同的公式中的一个得到（Fletcher，1987）：

1. Polak-Ribiére 公式，其中$\beta(n)$定义为

$$\beta(n) = \frac{\mathbf{r}^T(n)(\mathbf{r}(n)-\mathbf{r}(n-1))}{\mathbf{r}^T(n-1)\mathbf{r}(n-1)} \tag{4.134}$$

2. Fletcher-Reeves 公式，其中$\beta(n)$定义为

$$\beta(n) = \frac{\mathbf{r}^T(n)\mathbf{r}(n)}{\mathbf{r}^T(n-1)\mathbf{r}(n-1)} \tag{4.135}$$

为了用共轭梯度方法处理属于多层感知器无监督训练的代价函数$\mathscr{E}_{av}(\mathbf{w})$的无约束最优化问题，我们做两件事情：

- 用一个二次函数逼近代价函数$\mathscr{E}_{av}(\mathbf{w})$。也就是说，式(4.117)中三阶和更高阶项被忽略，这意味着我们正在逼近误差曲面上的一个局部最小值。在这个基础上，比较式(4.117)和式(4.122)，我们可以得到表4-7显示的联系。 239
- 用公式表示在共轭梯度算法中系数$\eta(n)$和$\beta(n)$的计算，使得仅仅要求梯度信息。

后面一点在多层感知器中特别重要，因为它避免使用Hessian矩阵$\mathbf{H}(n)$，该矩阵的估值是以计算上的困难著称的。

表 4-7　$f(\mathbf{x})$和$\mathscr{E}_{av}(\mathbf{w})$之间的对应

二次函数$f(\mathbf{x})$	代价函数$\mathscr{E}_{av}(\mathbf{w})$
参数向量$\mathbf{x}(n)$	突触权值向量$\mathbf{w}(n)$
梯度向量$\partial f(\mathbf{x})/\partial\mathbf{x}$	梯度向量$\mathbf{g}=\partial\mathscr{E}_{av}/\partial\mathbf{w}$
矩阵$\mathbf{A}$	Hessian 矩阵$\mathbf{H}$

没有Hessian矩阵$\mathbf{H}(n)$的明显知识时，为了计算决定搜索方向$\mathbf{s}(n)$的系数$\beta(n)$，我们可以利用式(4.134)的Polak－Ribiére公式或者式(4.135)中的Fletcher－Reeves公式。这两个

公式都仅包含残差的使用。假定一个二次函数，在共轭梯度方法的线性形式中，Polak - Ribiére 公式和 Fletcher - Reeves 公式是等价的。在另一方面，在非二次代价函数的情形下，它们不再等价。

对于非二次最优化问题，共轭梯度算法的 Polak-Ribiére 形式优先于该算法的 Fletcher-Reeves 式，针对这个问题我们在下面提供启发性的解释(Bertsekas, 1995)。由于代价公式 $\mathscr{E}_{av}(\mathbf{w})$中三阶与更高阶项存在和线搜索中可能的不精确性，所产生的搜索方向的共轭性逐渐丧失。这使得所产生的方向向量 $\mathbf{s}(n)$近似正交于残差 $\mathbf{r}(n)$的方向而算法可能会陷入"堵塞"。当这种现象出现的时候，我们有 $\mathbf{r}(n)=\mathbf{r}(n-1)$，在这种情况下标量 $\beta(n)$接近于零。相应地，方向向量 $\mathbf{s}(n)$近似于残差 $\mathbf{r}(n)$，从而打破堵塞。与此相反的是，当使用 Fletcher-Reeves 公式的时候，共轭梯度算法在相似的条件下显然继续堵塞。

然而，在极少数的情况下，Polak-Ribiére 方法可以无限循环下去而不收敛。值得庆幸的是，Polak-Ribiére 方法的收敛可以通过选择

$$\beta = \max\{\beta_{PR}, 0\} \tag{4.136}$$

得到保证(Shewchuk, 1994)，其中 β_{PR}是由式(4.134)的 Polak-Ribiére 公式定义的值。如果 $\beta_{PR}<0$，利用式(4.136)中定义的 β 的值等于重新开始共轭梯度算法。重新开始运算等于遗忘最后的搜索方向并且在最陡下降方向上重新开始(Shewchuk, 1994)。

考虑下一个计算参数 $\eta(n)$的问题，它决定共轭梯度算法的学习率。和计算 $\beta(n)$的一样，计算 $\eta(n)$的首选办法是避免必须使用 Hessian 矩阵 $\mathbf{H}(n)$。我们回忆基于式(4.126)的线最小化导出的 $\eta(n)$的公式和源于更新公式(4.125)得到的 $\eta(n)$计算公式的相同。因此我们需要一个*直线搜索*[16]，这样的目的是对 η 最小化函数$\mathscr{E}_{av}(\mathbf{w}+\eta\mathbf{s})$。也就是说，给定向量 $\mathbf{w}$ 和 $\mathbf{s}$ 的固定值，现在的问题是改变 η 使得函数最小化。随着 η 的变化，自变量 $\mathbf{w}+\eta\mathbf{s}$ 在 $\mathbf{w}$ 的 W 维向量空间中画出一条直线，因此称为"直线搜索"。*直线搜索算法*是一个迭代过程，它为共轭梯度算法的每次迭代产生一个估计序列$\{\eta(n)\}$。当找到令人满意的解时，直线搜索被停止。直线搜索必须在每个搜索方向上进行。

240

在文献中提出了几种直线索搜算方法，并且选择一个好的算法是重要地，因为它对被嵌入其中的共扼梯度法的性能具有深远的影响。任何直线索搜算法有两个阶段(Fletcher, 1987)：

- *包括阶段*，也就是搜索一段区间，即包含一个最小值的非平凡区间；
- *截段阶段*，在这个阶段中，区间被截成段(即被分割)，因此产生一系列长度越来越小的子区间。

现在我们叙述一个直接处理这两个阶段的曲线拟合过程。

令$\mathscr{E}_{av}(\eta)$表示多层感知器的代价函数，表示为 η 的函数。假设$\mathscr{E}_{av}(\eta)$是严格单峰的(unimodal)(即它在当前点 $\mathbf{w}(n)$的附近只有单一的最小值)并且是二次连续可微的。我们沿直线开始搜索过程，直到求出满足条件

$$\mathscr{E}_{av}(\eta_1) \geq \mathscr{E}_{av}(\eta_3) \geq \mathscr{E}_{av}(\eta_2) \qquad 对于\ \eta_1 < \eta_2 < \eta_3 \tag{4.137}$$

的三个点 η_1、η_2、η_3，如图 4-25 所示。由于$\mathscr{E}_{av}(\eta)$是 η 的连续函数，式(4.137)描述的选择保证区间$[\eta_1, \eta_3]$包含函数$\mathscr{E}_{av}(\eta)$的一个最小值。假设函数$\mathscr{E}_{av}(\eta)$充分光滑，我们可以认为这个函数在紧邻最小值的区间是抛物线形的。因此，我们可以使用反抛物线插值法(inverse

parabolic interpolation)进行分段(Press et al.,1988)。具体地，这个抛物线函数可以通过三个初始点 η_1、η_2、η_3拟合，如图4-26所示，图中实线对应于$\mathscr{E}_{av}(\eta)$，虚线表示分段过程的第一次迭代。令 η_4表示通过三点 η_1、η_2、η_3的抛物线的最小值点。在图4-26所示的例子中，我们有$\mathscr{E}_{av}(\eta_4)<\mathscr{E}_{av}(\eta_2)$，$\mathscr{E}_{av}(\eta_4)<\mathscr{E}_{av}(\eta_1)$。点 η_3由 η_4代替，作为新的区间$[\eta_1, \eta_4]$。通过构造一条通过点 η_1、η_2、η_4抛物线重复这个过程。上述包括区间后再分段的过程重复多次，直到找到一个足够接近$\mathscr{E}_{av}(\eta)$的最小值的点，此时直线搜索终止。 241

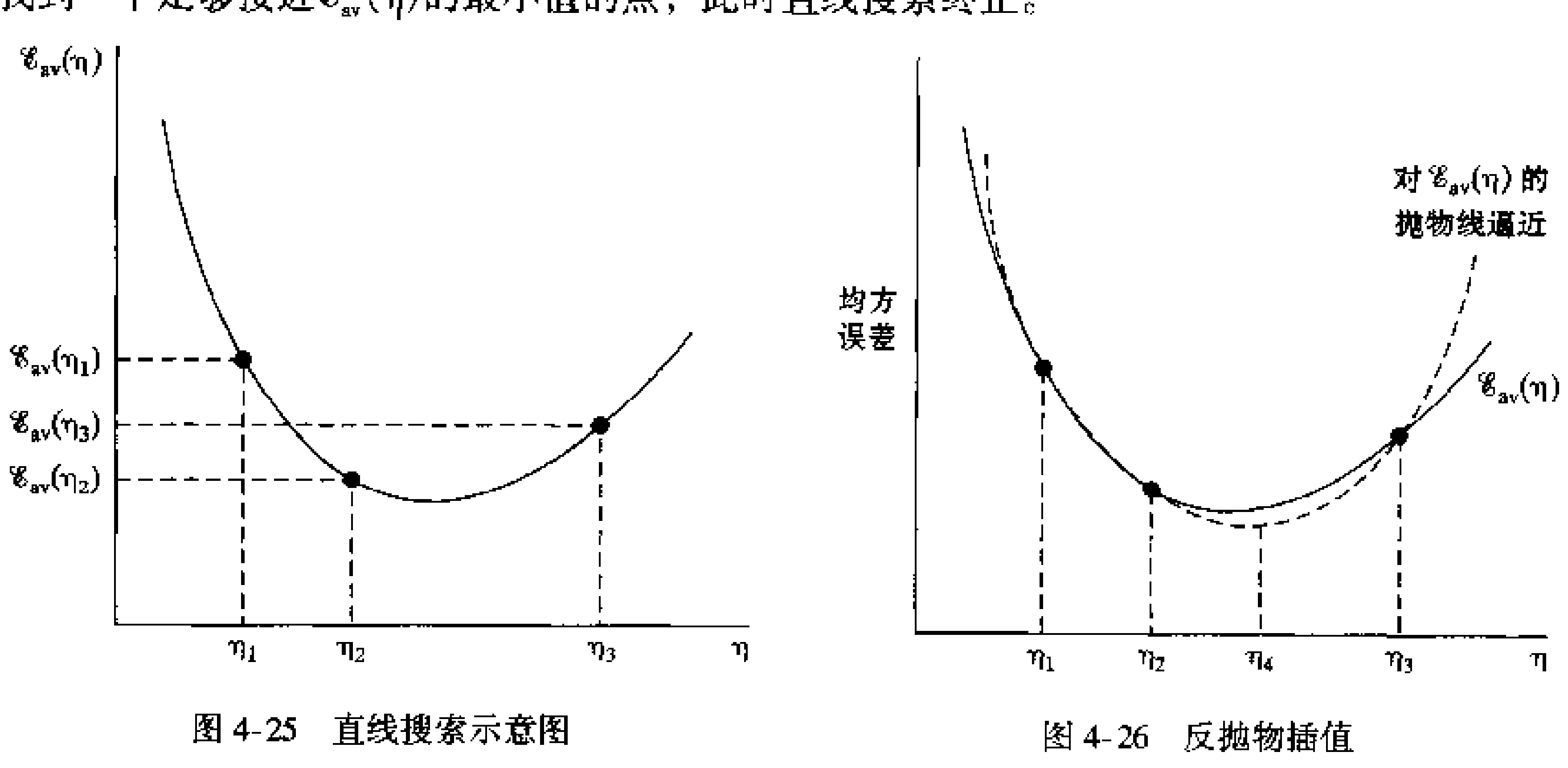


图4-25 直线搜索示意图

图4-26 反抛物插值

Brent的方法建立刚才所述的三点曲线拟合过程的一个高度精练的形式(Press et al., 1988)。在计算的任何特殊阶段，Brent方法保持$\mathscr{E}_{av}(\eta)$函数六个点的轨迹，所有点可能不必互不相同。如前所述，抛物线插值试图通过这些点中的三个。为了使得这个插值法是可接受的，剩下的三点必须满足一定标准。最终结果是一个鲁棒直线搜索算法。

非线性共扼梯度算法小结

现在我们给出形式描述用于多层感知器监督训练的共扼梯度算法的非线性(非二次)形式的所有需要的要素。表4-8给出该算法的小结。

拟Newton方法

重新开始讨论拟Newton方法，我们发现这些基本上是梯度方法，用更新公式

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta(n)\mathbf{s}(n) \tag{4.138}$$

表示，其中方向向量 $\mathbf{s}(n)$用梯度向量 $\mathbf{g}(n)$定义为

$$\mathbf{s}(n)=-\mathbf{S}(n)\mathbf{g}(n) \tag{4.139}$$

矩阵$\mathbf{S}(n)$是在每次迭代中调整的正定矩阵。这样做是为了使得方向向量 $\mathbf{s}(n)$逼近牛顿方向，即

$$-(\partial^2\mathscr{E}_{av}/\partial\mathbf{w}^2)^{-1}(\partial\mathscr{E}_{av}/\partial\mathbf{w})$$

242

拟Newton方法使用误差曲面的二阶(曲率)信息，实际上不要求Hessian矩阵$\mathbf{H}$的知识。这通过使用两次连续迭代 $\mathbf{w}(n)$、$\mathbf{w}(n+1)$与梯度向量 $\mathbf{g}(n)$、$\mathbf{g}(n+1)$来实现。令

表 4-8 用于多层感知器有监督训练的非线性共轭梯度算法小结

初始化

除非权值向量 $\mathbf{w}$ 的先验知识是可用的，否则使用与反向传播算法相似的过程选择初始值 $\mathbf{w}(0)$。

计算

1. 对于 $\mathbf{w}(0)$，用反向传播算法计算梯度向量 $\mathbf{g}(0)$。
2. 设置 $\mathbf{s}(0)=\mathbf{r}(0)=-\mathbf{g}(0)$。
3. 在时间步 n，用直线搜索寻找充分最小化 $\mathscr{E}_{av}(\eta)$ 的 $\eta(n)$，对于固定的 $\mathbf{w}$ 和 $\mathbf{s}$，代价函数 $\mathscr{E}_{av}$ 表示为 η 的函数。
4. 测试决定 $\mathbf{r}(n)$ 的欧几里德范数是否下降到一个特定的值之下，即为初始值 $\|\mathbf{r}(0)\|$ 的很小的一部分。
5. 更新权值向量：
$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta(n)\mathbf{s}(n)$$
6. 对于 $\mathbf{w}(n+1)$，用反向传播算法计算更新的梯度向量 $\mathbf{g}(n+1)$。
7. 设置 $\mathbf{r}(n+1)=-\mathbf{g}(n+1)$。
8. 用 Polak-Ribiére 方法计算 $\beta(n+1)$：
$$\beta(n+1)=\max\left\{\frac{\mathbf{r}^T(n+1)(\mathbf{r}(n+1)-\mathbf{r}(n))}{\mathbf{r}^T(n)\mathbf{r}(n)},0\right\}$$
9. 更新方向向量：
$$\mathbf{s}(n+1)=\mathbf{r}(n+1)+\beta(n+1)\mathbf{s}(n)$$
10. 设置 $n=n+1$，转第3步。

停止准则　当下述条件满足时结束算法：
$$\|\mathbf{r}(n)\|\leqslant\varepsilon\|\mathbf{r}(0)\|$$
其中 ε 是一个指定的小数。

$$\mathbf{q}(n)=\mathbf{g}(n+1)-\mathbf{g}(n)\tag{4.140}$$
和
$$\Delta\mathbf{w}(n)=\mathbf{w}(n+1)-\mathbf{w}(n)\tag{4.141}$$
这样我们可以通过逼近式
$$\mathbf{q}(n)\simeq\left(\frac{\partial}{\partial\mathbf{w}}\mathbf{g}(n)\right)\Delta\mathbf{w}(n)\tag{4.142}$$
得到曲率信息。特别地，给定 W 个线性独立的权值增量 $\Delta\mathbf{w}(0)$，$\Delta\mathbf{w}(1)$，…，$\Delta\mathbf{w}(W-1)$ 和各自的梯度增量 $\mathbf{q}(0)$，$\mathbf{q}(1)$，…，$\mathbf{q}(W-1)$，我们可以逼近 Hessian 矩阵 $\mathbf{H}$ 如下：

243

$$\mathbf{H}\simeq[\mathbf{q}(0),\mathbf{q}(1),\cdots,\mathbf{q}(W-1)][\Delta\mathbf{w}(0),\Delta\mathbf{w}(1),\cdots,\Delta\mathbf{w}(W-1)]^{-1}\tag{4.143}$$
我们也可以逼近逆 Hessian 矩阵如下：
$$\mathbf{H}^{-1}\simeq[\Delta\mathbf{w}(0),\Delta\mathbf{w}(1),\cdots,\Delta\mathbf{w}(W-1)][\mathbf{q}(0),\mathbf{q}(1),\cdots,\mathbf{q}(W-1)]^{-1}\tag{4.144}$$
当代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 为二次函数的时候，式(4.143)和(4.144)是精确的。

在最常用的一类拟 Newton 方法中，矩阵 $\mathbf{S}(n+1)$ 由它先前的值 $\mathbf{S}(n)$，向量 $\Delta\mathbf{w}(n)$ 和 $\mathbf{q}(n)$ 三项使用递归算式得到(Fletcher,1987;Bertsekas,1995)：
$$\mathbf{S}(n+1)=\mathbf{S}(n)+\frac{\Delta\mathbf{w}(n)\Delta\mathbf{w}^T(n)}{\mathbf{q}^T(n)\mathbf{q}(n)}-\frac{\mathbf{S}(n)\mathbf{q}(n)\mathbf{q}^T(n)\mathbf{S}(n)}{\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)}+\xi(n)[\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)][\mathbf{v}(n)\mathbf{v}^T(n)]\tag{4.145}$$
其中
$$\mathbf{v}(n)=\frac{\Delta\mathbf{w}(n)}{\Delta\mathbf{w}^T(n)\Delta\mathbf{w}(n)}-\frac{\mathbf{S}(n)\mathbf{q}(n)}{\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)}\tag{4.146}$$
并且
$$0\leqslant\xi(n)\leqslant1\qquad 对于所有\ n\tag{4.147}$$
该算法由任意定义的正定矩阵 $\mathbf{S}(0)$ 进行初始化。拟 Newton 方法的特殊形式参数化为如何定

义标量 $\eta(n)$，如下所示(Fletcher, 1987)：

- 对于所有 n 满足 $\xi(n)=0$，我们得到 Davidon-Fletcher-Powell(DFP)算法，它是历史上最初的拟 Newton 方法。
- 对于所有 n 满足 $\xi(n)=1$，我们得到 Broyden-Fletcher-Goldfarb-Shanno 算法，它在目前被认为是拟 Newton 方法的最好形式。

拟 Newton 方法和共扼梯度法的比较

我们通过在非二次最优化问题背景下对拟 Newton 方法和共扼梯度法的比较，来结束拟 Newton 方法的简要讨论(Bertsekas, 1995)：

- 拟 Newton 方法和共扼梯度法都避免使用 Hessian 矩阵。然而，拟 Newton 方法通过逼近逆 Hessian 矩阵来进行下一步计算。所以，当直线搜索是精确的并且充分逼近一个具有正定 Hessian 矩阵的局部最小值时，拟 Newton 方法趋于逼近 Newton 方法，因此得到的收敛速度比共扼梯度法可能的收敛速度更快。
- 拟 Newton 方法对在最优化的直线搜索阶段精度的灵敏性不如共扼梯度法。
- 除了方向向量 $\mathbf{S}(n)$ 计算相关的矩阵向量乘法之外，拟 Newton 方法还要求存储矩阵 $\mathbf{S}(n)$。最后结果是拟 Newton 方法的计算复杂度是 $O(W^2)$。相反，共扼梯度法的计算复杂度为 $O(W)$。这样，当维数 W(即权值向量 $\mathbf{w}$ 的个数)很大时，共扼梯度法比拟 Newton 方法在计算上具有更大的优越性。 244

正是因为后面这一点，实际上拟 Newton 方法限于小规模神经网络的设计。

4.19 卷积网络

到目前为止，我们都在考虑多层感知器算法设计和相关的问题。本节我们集中在多层感知器本身的结构布局问题上。特别地，我们描述一类特定的通称为*卷积网络*的多层感知器；这些网络所隐含的思想已经在第 1 章简要给出。

一个*卷积网络*是为识别二维形状而特殊设计的一个多层感知器，这种二维形状对平移、比例缩放、倾斜或者其他形式的变形具有高度不变性。这个艰巨的任务是通过如下网络在监督方式下学会的，网络的结构包括如下形式的约束(LeCun and Bengio, 1995)：

1. *特征提取*。每一个神经元从上一层的局部接受域得到突触输入，因而迫使它提取局部特征。一旦一个特征被提取出来，只要它相对于其他特征的位置被近似地保留下来，它的精确位置就变得没有那么重要了。

2. *特征映射*。网络的每一个计算层都是由多个特征映射组成的，每个特征映射都是平面形式的，平面中单独的神经元在约束下共享相同的突触权值集。这种结构约束的第二种形式具有如下的有益效果：

- *平移不变性*，强迫特征映射的执行使用具有小尺度核的卷积，再接着用一个 sigmoid (挤压)函数。
- *自由参数数量的缩减*，通过权值共享实现。

3. *子抽样*。每个卷积层跟着一个实现局部平均和子抽样的计算层，由此特征映射的分辨率降低。这种操作具有使特征映射的输出对平移和其他形式的变形的敏感度下降的作用。

正如所述，卷积网络的发展是由神经生物学激发的，这可追溯到 Hubel 和 Wiesel(1962,

1977)关于猫的视觉皮层上局部灵敏和方位选择神经元的开拓性工作。

[245] 我们强调指出在一个卷积网络所有层中的所有权值都是通过训练来学习的。此外，网络自动地学习提取它自身的特征。

图4-27表明由一个输入层和四个隐藏层与一个输出层组成的卷积网络的体系结构布局。这个网络被设计用于实现图像处理(例如手写体的识别)。输入层由 28×28 个感知节点组成，接收已经近似处于中心位置和在大小上规整化的不同字符的图像。然后，计算流程在卷积和子抽样之间交替，如下所述：

- 第一隐藏层进行卷积。它由四个特征映射组成，每个特征映射由 24×24 个神经元组成。每个神经元指定一个 5×5 的接受域；
- 第二隐藏层实现子抽样和局部平均。它同样由四个特征映射组成，但其每个特征映射由 12×12 个神经元组成。每个神经元具有一个 2×2 的接受域，一个可训练系数，一个可训练偏置和一个 sigmoid 激活函数。可训练系数和偏置控制神经元的操作点；例如，如果系数很小，该神经元以拟线性方式操作。
- 第三隐藏层进行第二次卷积。它由12个特征映射组成，每个特征映射由 8×8 个神经元组成。该隐藏层中的每个神经元可能具有和上一个隐藏层几个特征映射相连的突触连接。否则，它以第一个卷积层相似的方式操作。
- 第四个隐藏层进行第二次子抽样和局部平均计算。它由12个特征映射组成，但每个特征映射由 4×4 个神经元组成。否则它以第一次抽样相似的方式操作。
- 输出层实现卷积的最后阶段。它由26个神经元组成，每个神经元指定为26个可能的字符中的一个。跟前面一样，每个神经元指定一个 4×4 的接受域。

相继的计算层在卷积和抽样之间的连续交替，我们得到一个"双尖塔"的效果。也就是在每个卷积或抽样层，随着空间分辨率下降，与相应的前一层相比特征映射的数量增加。卷积[246]之后进行子抽样的思想是受到 Hubel 和 Wiesel(1962)首先描述的"简单的"细胞后面跟着"复杂的"细胞[17]的想法的启发而产生的。

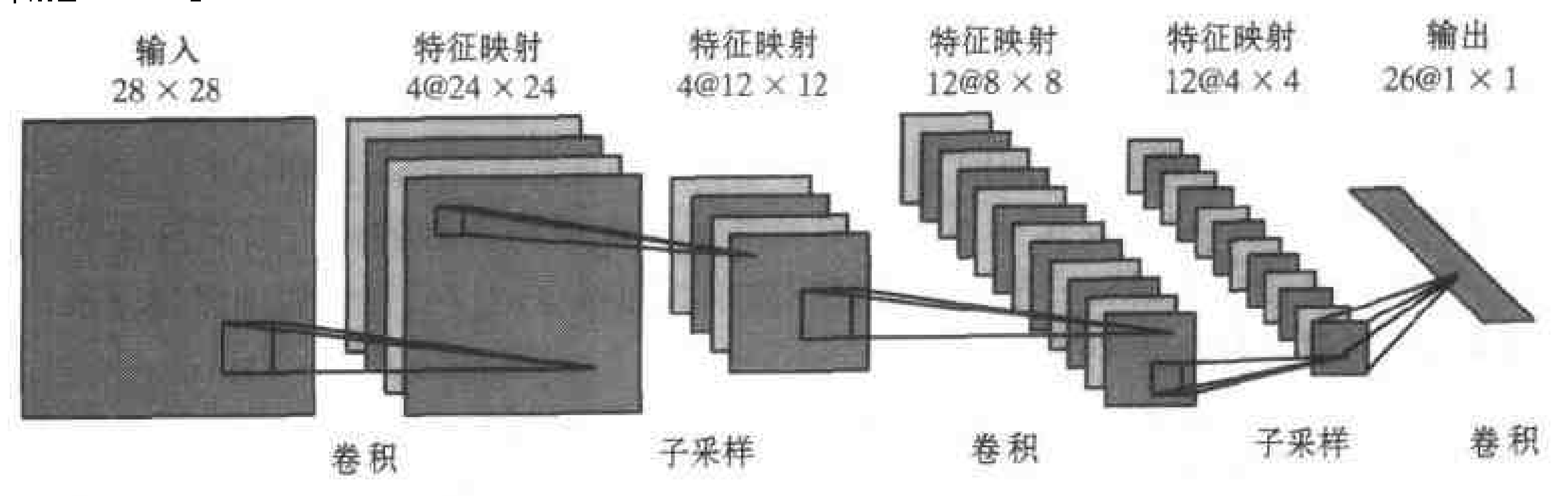


图4-27 用于图像处理如手写体识别的卷积网络(MIT出版社允许复制)

图4-27所示的多层感知器包含近似100 000个突触连接，但只有大约2 600个自由参数。自由参数在数量上显著地减少是通过权值共享获得的。学习机器的能力(以VC维的形式度量)因而下降，这又提高它的泛化能力(LeCun,1989)。甚至更值得注意的是对自由参数的调整通过反向传播学习的随机(串行的)形式来实现。

另一个显著的特点是使用权值共享使得以并行形式实现卷积网络变得可能。这是卷积网络对完全连接的多层感知器而言的另一个优点。

从图 4-27 的卷积网络中学习的经验有两个方面。首先，通过结合当前任务的先验知识约束其设计，一个易调整大小的多层感知器能够学习一个复杂的、高维的和非线性的映射。其次，突触权值和偏置水平可以周而复始地执行通过训练集的简单反向传播算法进行学习。

4.20 小结和讨论

反向传播学习已经成为多层感知器的训练的标准算法，它通常作为其他学习算法的基准。反向传播算法的名字起源于这样一个事实，网络的代价函数(性能度量)对自由参数(突触权值和偏置)的偏导数是由通过网络一层一层反向传播误差信号(由输出神经元计算)所决定的。在这样的处理过程中，它以非常高明的方式解决信任赋值(credit-assignment)的问题。该算法的计算能力在于它的两个主要特征：

- 更新多层感知器突触权值和偏置的局部方法。
- 计算代价函数对这些自由参数的所有偏导数的高效方法。

对于训练数据的一个给定回合，反向传播算法以这样两个方式中的一个操作：串行的方式或者集中式的方式。在串行方式中网络的所有神经元的突触权值都是在一个模式接着一个模式的基础上调整的。因此，在计算中使用的误差曲面梯度向量的估算值在本质上是随机的(任意的)，因此"随机反向传播"的名称同样是用来指反向传播学习的串行方式。在另一方面，在集中式方式中，对所有突触权值和偏置的调整是在一个回合接一个回合的基础上进行的，这样在计算中使用梯度向量更精确的估计。无论它的缺点如何，反向传播学习的串行(随机)形式是神经网络设计中使用频率最高的，特别是在大型问题上。为了得到最好的结果，需要小心地调整算法。 247

在多层感知器设计中的特定细节问题自然依赖于有关具体的应用。然而，我们可以做出两种区分：

1. 在涉及非线性可分模式的模式分类中，网络中的所有神经元都是非线性的。这个非线性是通过使用 sigmoid 函数来获得的，该函数的两种通常用法是(a)非对称 logistic 函数，和(b)反对称双曲正切函数。每个神经元负责在决策空间中产生它自己的超平面。通过一个监督的学习过程，网络中由所有神经元形成的超平面的组合被反复调整，使之分离来自不同类的以前未曾见过的模式时具有最少的平均分类误差。对于模式分类来说，随机反向传播算法是实现训练最广泛使用的算法，特别是在大型问题上(例如光学字符识别)。

2. 在非线性回归中，多层感知器的输出范围应该大到足以以包含过程值；如果这个信息不能得到，那么线性输出神经元的使用是最明智的选择。对学习算法，我们提供如下的观察事实：

- 反向传播学习的串行(随机)方式比集中方式慢得多。
- 反向传播学习集中方式比共扼梯度方法慢。然而，注意后一种方法只能在集中方式中使用。

我们以一些关于性能度量的最后评论结束这一讨论。本章中提出的反向算法的推导是基于以这种或那种方法最小化代价函数$\mathscr{E}_{av}$，代价函数$\mathscr{E}_{av}$定义为误差平方和在整个训练集上平均。这个准则的一个重要优点是它的普遍性和数学上的易处理性。然而，实际中遇到的许多情况，最小化代价函数$\mathscr{E}_{av}$相当于优化并不是系统最终目标的中间量，并且可能因此导致一个次优的性能。例如，在资本市场交易系统中，一个投资者或交易者的最终目标是以最小的

风险获得最大的预期回报(Choey and Weigend,1996;Moody and Wu,1996)。作为风险调整回报的性能评价标准的夏普率(Sharpe ratio)或回报易失率(reward-to-volatility ratio)从直觉上比$\mathscr{E}_{av}$更有吸引力。

注释和参考文献

[1] sigmoid函数被这样命名是因为它们的图形是"s"形的。Menon et al.(1996)对两类sigmoid函数进行了深入的研究:

- 简单sigmoid,定义为渐进有界的和完全单调的单变量奇函数。
- 双曲sigmoid,代表简单sigmoid的一个真子集和双曲线正切函数的自然推广。

248

[2] 对于LMS算法的特殊情形,已经证明使用动量常数α降低学习率参数η的稳定范围,并且如果η没有被适当调整,这样会导致不稳定。此外,错误调整也随α的增加而增长;更详细的论述请见Roy and Shynk(1990)。

[3] 对于从第一条原则中导出包含动量常数的反向传播算法,见Hagiwara(1992)。

[4] 如果向量$\mathbf{w}^*$不比它邻近的点向量更差的话,向量$\mathbf{w}^*$被称为输入输出函数F的一个局部最小值;也就是,如果存在一个ε如下(Bertsekas,1995):

$$F(\mathbf{w}^*) \leqslant F(\mathbf{w}) \qquad 对所有满足 \|\mathbf{w}-\mathbf{w}^*\| < \varepsilon 的 \mathbf{w}$$

如果$\mathbf{w}^*$不比其他所有的向量都差,则称它为函数F的一个全局最小值;也就是,

$$F(\mathbf{w}^*) \leqslant F(\mathbf{w}) \qquad 对所有的 \mathbf{w} \in \mathbb{R}^n$$

其中n是$\mathbf{w}$的维数。

[5] 对有效梯度估计应用反向传播的首次文献记载应归功于Werbos(1974)。在4.10节中给出的材料依照Saarinen et al.(1992)给出的处理方法;Werbos(1990)对该题目给出更一般的讨论。

[6] 网络设计得益于Hessian矩阵知识的其他方面包括(Bishop,1995):

(1)在训练数据中进行很小变化后,Hessian矩阵组成多层感知器再训练过程的基础。

(2)在Bayes学习的背景下:

- Hessian矩阵的逆可用于为训练后的神经网络作出的非线性预测提供误差条,并且
- Hessian矩阵的特征值可以用于决定正则化参数的合适值。

[7] Buntine和Weigend(1994)回顾计算Hessian矩阵的精确算法和近似算法,并有特别针对神经网络的参考文献;也可参考Battiti(1992)的文章。

[8] 通用逼近定理可以看作是Weierstrass定理(Weierstrass,1885)的自然扩展。这个定理表明任何一个在实轴闭区间上的连续函数都可以表示成该区间上绝对一致收敛的多项式级数的极限。

以多层感知器作为工具进行对任意连续函数表示的研究很可能是首先被

249

HechtNielsen(1987)提起关注,他引用了归功于Sprecher(1965)的Kolomogorov叠加定理的改进版本。然后Gallant和White(1988)证明,在隐藏层具有单调"余弦"挤压和在输出无挤压的单隐藏层多层感知器是被作为"Fourier网络"的特殊情形嵌入的,它的输出产生给定函数的Fourier级数逼近。然而,在传统的多层感知器背景下,Cybenko第一次严格证明了一个隐藏层足够一致逼近任何具有在单位超立方体中的支集的函数;这项工作作为1988伊利诺斯大学的技术报告发表,一年之后作为论文发表(Cybenko,1988,

1989)。在 1989 年，另外两篇关于多层感知器通用逼近器的论文独立发表了，一篇由 Funahashi 完成，另外一篇由 Hornik，Stinchcombe 和 White 完成。对后来关于逼近问题的贡献，请见 Light(1992b)。

[9] 交叉确认的发展历史在 Stone(1974)中有记载。交叉确认的思想至少在 20 世纪 30 年代就已广泛传播，但该项技术的改进是在 20 世纪 60 年代和 70 年代完成的。该领域的两篇重要论文是 Stone(1974)和 Geisser(1975)，他们独立地并且几乎同时提出这项技术。这项技术被 Stone 命名为“交叉确认方法”，而 Geisser 则称之为“预测样本复用方法”。

[10] 关于训练早期停止方法的最初参考文献包括 Morgan and Bourlard(1990)和 Weigend et al.(1990)。也许对多层感知器训练早期停止方法最详尽的统计学分析是由 Amari et al.(1996a)提出的。这项研究得到具有 108 个可调整参数和一个非常巨大的数据集(50 000 个样本)的 8-8-4 分类器的计算机仿真的支持。

[11] *级联相关学习体系结构*(Fahlman and Lebiere, 1990)是网络生长方法的一个例子。该过程从一个最小网络开始，这个最小网络具有基于输入/输出考虑而指定的一些输入和一个或者更多的输出节点，但隐藏层没有节点。例如，LMS 算法可以用来训练网络。隐藏神经元被一个接一个地添加到网络中，因此得到一个多层结构。每个新的神经元从每个输入节点接受一个突触连接，并且从每个先前存在的隐藏神经元同样接受连接。当增加一个新的隐藏神经元的时候，该神经元输入边的突触连接被冻结；只有在输出边的突触连接被反复地训练。这个被加进去的隐藏神经元就成为网络中永久的特征检测器。添加新的隐藏神经元的过程如上述形式进行直到得到令人满意的性能为止。

然而在 Lee et al.(1990)所论述的网络生长方法中，在前向通过(函数级自适应)和反向通过(参数级自适应)上增加了称为*结构级自适应*的第三级计算。在第三级计算中，网络的结构通过改变神经元的数量和网络中神经元之间的结构关系而进行调整。这里所使用的准则是当估计误差(收敛之后)比期望的值大，则在网络中最需要的地方增加一个神经元。新的神经元的合适位置取决于监督网络的学习行为。特别地，如果在一个长期的参数调整(训练)之后，某神经元输入的突出连接权值向量连续显著地波动，可以推断正被讨论的神经元没有足够的表达能力学习它所承担的任务。结构级自适应同样包括防备神经元可能出现的灭绝。一个神经元当它不在是网络的功能元素或者它是网络中多余元素的时候，它将灭绝。这种网络增长的方法看起来是计算密集的。

[12] Hecht-Nielsen(1995)描述一种复制器神经网络，它是具有三个隐藏层和一个输出层的多层感知器的形式：

- 在第二和第四(隐含)层中的激活函数通过双曲正切函数定义：

$$\varphi^{(2)}(v) = \varphi^{(4)}(v) = \tanh(v)$$

其中 v 是在这些层中一个神经元的被包含的诱导局部域。 250

- 在中间(隐含)层的每个神经元的激活函数由

$$\varphi^{(3)}(v) = \frac{1}{2} + \frac{1}{2(N-1)}\sum_{j=1}^{N-1}\tanh\left(a\left(v - \frac{j}{N}\right)\right)$$

给出，其中 a 是一个增益参数，v 是该层中神经元的诱导局部域。函数 $\varphi^{(3)}(v)$ 描述一个光滑的具有 N 级的阶梯激活函数，因而本质把相关神经元层的输出向量转化为 $K = N^n$ 级，其中 n 是中间隐藏层的神经元数目。

- 输出层中的神经元是线性的，它们的激活函数定义为

$$\varphi^{(5)}(v) = v$$

基于这种神经网络结构，Hecht-Nielsen 提出了一个定理，证明对随机输入数据向量的最佳数据压缩是可以得到的。

[13] 我们最起码需要是一个解释局部最小问题的反向传播学习的理论框架。这是一个难以完成的任务。不过，在文献中已有关于这个问题的一些进展的报告。Baldi 和 Hornik (1989)考虑了具有线性激活函数的分层前馈神经网络使用反向传播学习中的学习问题。他们论文中的主要结论是误差曲面只有惟一的最小值，对应于训练模式的协方差矩阵第一主特征向量所扩张的子空间上的正交投影；误差曲面上所有的其他临界点都是鞍点。Gori 和 Tesi(1992)考虑了反向传播更一般的情形，包括使用非线性神经元。他们论文中的主要结论是对于线性可分模型，可以通过使用反向传播学习的集中处理方式来确保收敛于一个最优解(也就是全局最小值)，并且网络对新样本的泛化能力超过了 Rosenblatt 模型。

[14] 基于启发 1 到启发 4 对反向传播算法的修改被称为 delta-bar-delta 学习规则(Jacobs, 1988)，它来源于与在 4.3 节导出反向传播算法的传统形式相似的过程。delta-bar-delta 学习规则的实现可以通过采用与梯度复用方法(Hush and Sales, 1988; Haykin and Deng, 1991)相似的思想来进行简化。

Salomon 和 Van Hemmen(1996)提出一种加速反向传播学习过程的动态自适应过程。它的根本思想是用前一时间步的学习率，轻微地增加和减少它，对学习率参数的这两个新的值求代价函数的值，然后选择使代价函数取值小的一个。

[15] 共轭梯度方法的经典参考文献是 Hestenes and Stiefel(1952)的著作。关于共轭梯度算法收敛行为的讨论，见 Luenberger(1984) and Bertsekas(1995)。关于共轭梯度算法的许多方面的指导性处理方法，见 Shewchuk(1994)。关于在神经网络领域中该算法的易读文献见 Johansson et al.(1990)。

[16] 共轭梯度算法的传统形式要求使用直线搜索方法，它可能因为自身的尝试性和误差性而花费时间。Møller(1993)描述共轭梯度算法的一个修改版本，称为比例共轭梯度算法，它避免使用直线搜索。从本质上来说，直线搜索由算法的一维空间的 Levenberg-Marquardt 形式代替。使用这种办法的动机是避开由非正定 Hessian 矩阵引起的困难(Fletcher, 1987)。

[17] Hubel 和 Wiesel 关于"简单"和"复杂"细胞的概念在神经网络文献中第一次被 Fukushima (1980, 1995)在设计一个称为神经认知机的学习机的过程中所利用。然而，这个学习机以自组织的形式运行，而图 4-27 描述的卷积网络使用标定的样本以监督的形式运行。

251

习题

XOR 问题

4.1 为了解决 XOR 问题，图 4-28 表示一个包括单个隐藏神经元的神经网络；这个网络可以看作是在 4.5 节中所考虑的替代模型。通过构建(a)决策区域和(b)网络的真值表，证明图 4-28 表示的网络

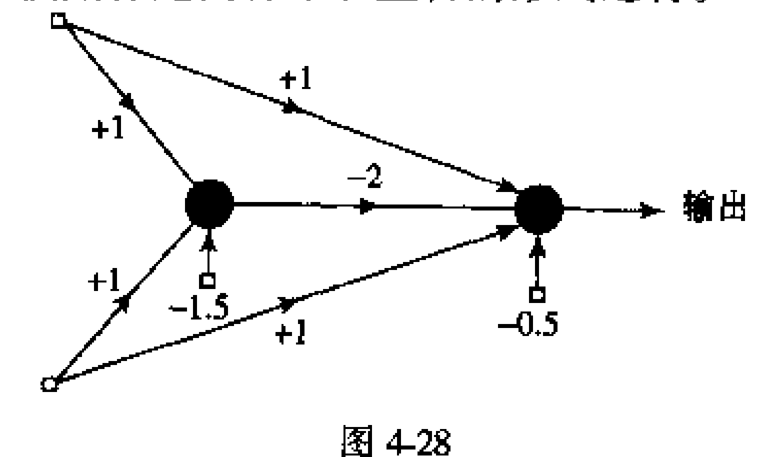


图 4-28

解决了 XOR 问题。

4.2 使用反向传播算法为图 4-8 所示的神经网络计算一组突触权值和偏置的值以解决 XOR 问题。假设非线性使用一个 logistic 函数。

反向传播学习

4.3 在权值更新中包含的动量项可以认为是满足启发 3 和 4 的机制，它们为加速反向传播算法的收敛提供指导，这在 4.17 节中进行了讨论。说明这个陈述的正确性。

4.4 动量项 α 通常被指定为在 $0 \leqslant \alpha < 1$ 范围的正值。如果 α 是赋予在 $-1 < \alpha \leqslant 0$ 之间的一个负值，研究在这样的条件下使得式(4.41)关于时间 t 的行为差异。

4.5 考虑包括单个权值的网络的简单例子，它的代价函数是

$$\mathscr{E}(w) = k_1(w - w_0)^2 + k_2$$

其中 w_0、k_1 和 k_2 是常数。用具有动量项的反向传播算法最小化$\mathscr{E}(w)$。

探索包含的动量项常数 α 怎样影响学习过程。特别注意使用 α 收敛所需的步数。

4.6 在 4.7 节中我们给出了多层感知器分类器（非线性性使用 logistic 函数）属性的定性分析，它的输出提供后验分类概率的估计。这个性质假设训练集足够大，并且用来训练网络的反向传播算法不会在一个局部最小上被阻塞。补充这个性质的数学细节。

4.7 从式(4.70)所定义的代价函数开始，推导式(4.72)的最小化解和式(4.73)定义的代价函数的最小值。

4.8 式(4.81)到(4.83)定义图 4-18 中的多层感知器实现的逼近函数 $F(\mathbf{w},\mathbf{x})$的偏导数，根据如下的假设推导这些公式：

(a)代价函数：

$$\mathscr{E}(n) = \frac{1}{2}[d - F(\mathbf{w},\mathbf{x})]^2$$

252

(b)神经元 j 的输出：

$$y_j = \varphi\left(\sum_i w_{ji} y_i\right)$$

其中 w_{ji}是从神经元 i 到神经元 j 的突触权值，y_i 是神经元 i 的输出；

(c)非线性性；

$$\varphi(v) = \frac{1}{1 + \exp(-v)}$$

交叉确认

4.9 在第 2 章所讨论的结构风险最小化的研究中，也许会说交叉确认是其中的一种情形。描述一个使用交叉确认的神经网络的例子，支持这个说法。

4.10 在多重交叉确认中并没有如坚持到底方法中那样在训练数据和测试（确认）数据之间有明确的区分。使用多重交叉确认可能产生有偏估计吗？证明你的答案。

网络修剪技术

4.11 模型选择的统计学准则，如 Rissanen 最小描述长度(MDL)准则和 Akaike 的信息论原则(AIC)，共用一个常用的组成形式：

(模型复杂度准则) = (对数似然函数) + (模型复杂度惩罚)

讨论用于网络修剪的权值衰减和权值消除方法是如何符合这种形式的。

4.12 (a)推导式(4.105)给出的显著性 S_i 的公式；

(b)假设多层感知器的均方误差对自身权值的 Hessian 矩阵可以被对角阵

$$\mathbf{H} = \mathrm{diag}[h_{11}, h_{22}, \cdots, h_{WW}]$$

逼近，其中 W 是网络权值的总数。决定网络中权值 w_i 的显著性 S_i。

反向传播学习的加速收敛

4.13 delta-bar-delta 学习规则(Jacobs, 1988)代表反向传播算法的一个修改形式，它基于 4.17 节中所述的启发。在这个规则中，网络中的每个突触权值被指定一个自身的学习率参数。代价函数 $E(n)$因而以相应的方式中被修改。换句话说，尽管 $E(n)$在数学上是相似于式(4.2)的代价函数$\mathscr{E}(n)$的，但是新的代价函数 $E(n)$的参数空间包括不同的学习率。

(a)推导偏导数$\partial E(n)/\partial \eta_{ji}(n)$的表达式，其中 $\eta_{ji}(n)$为相应于突触权值 $w_{ji}(n)$的学习率参数。

253

(b)因此，说明基于(a)的结果的学习率参数调整是完全符合 4.17 节中启发 3 和启发 4 的。

二阶最优化方法

4.14 在式(4.39)所述的权值修改中动量项的使用可以被认为是共轭梯度方法的近似(Battiti, 1992)。讨论这种说法的正确性。

4.15 以式(4.133)中 $\beta(n)$的公式开始，推导 Hesteness-Stiefel 公式

$$\beta(n) = \frac{\mathbf{r}^T(n)(\mathbf{r}(n) - \mathbf{r}(n-1))}{\mathbf{s}^T(n-1)\mathbf{r}(n-1)}$$

其中 $\mathbf{s}(n)$是方向向量，$\mathbf{r}(n)$是共轭梯度方法中的余项。利用这个结果，推导式(4.134)中的 Polak-Ribiére 公式和式(4.135)中的 Fletcher-Reeves 公式。

计算机实验

4.16 研究使用 sigmoid 非线性函数的反向传播学习方法获得一对一映射，描述如下：

1. $f(x) = \dfrac{1}{x}$, $\quad 1 \leqslant x \leqslant 100$
2. $f(x) = \log_{10} x$, $\quad 1 \leqslant x \leqslant 10$
3. $f(x) = \exp(-x)$, $\quad 1 \leqslant x \leqslant 10$
4. $f(x) = \sin x$, $\quad 0 \leqslant x \leqslant \dfrac{\pi}{2}$

对每个映射，完成如下工作：

(a)建立两个数据集，一个用于网络训练，另一个用于测试。

(b)假设具有单个隐藏层，利用训练数据集计算网络的突触权值。

(c)通过使用测试数据求网络计算精度的值。

使用单个隐藏层，但隐藏神经元数目可变，研究网络性能是如何受隐藏层大小变化影响的。

4.17 表 4-9 的数据表示澳大利亚野兔眼睛晶状体的重量为年龄的函数。没有简单的解析函数可以精确插值这些数据，因为我们并没有一个单值函数。相反，利用一个负指数我们有这个数据集的一个非线性最小平方模型，表示为

254

$$y = 233.846(1 - \exp(-0.006042x)) + \varepsilon$$

其中 ε 是误差项。

利用反向传播算法，设计一个多层感知器，它能够为这个数据集提供一个非线性最小平方逼近。与前述的最小平方模型比较你的结果。

表 4-9 澳大利亚野兔眼睛晶状体重量

年龄（天）	重量（mg）	年龄（天）	重量（mg）	年龄（天）	重量（mg）	年龄（天）	重量（mg）
15	21.66	75	94.6	218	174.18	338	203.23
15	22.75	82	92.5	218	173.03	347	188.38
15	22.3	85	105	219	173.54	354	189.7
18	31.25	91	101.7	224	178.86	357	195.31
28	44.79	91	102.9	225	177.68	375	202.63
29	40.55	97	110	227	173.73	394	224.82
37	50.25	98	104.3	232	159.98	513	203.3
37	46.88	125	134.9	232	161.29	535	209.7
44	52.03	142	130.68	237	187.07	554	233.9
50	63.47	142	140.58	246	176.13	591	234.7
50	61.13	147	155.3	258	183.4	648	244.3
60	81	147	152.2	276	186.26	660	231
61	73.09	150	144.5	285	189.66	705	242.4
64	79.09	159	142.15	300	186.09	723	230.77
65	79.51	165	139.81	301	186.7	756	242.57
65	65.31	183	153.22	305	186.8	768	232.12
72	71.9	192	145.72	312	195.1	860	246.7
75	86.1	195	161.1	317	216.41		

255

第5章　径向基函数网络

5.1　简介

设计一个监督神经网络可以有多种方法。前面一章中所描述的反向传播算法可以看作是递归技术的应用，这种技术在统计学中通称为*随机逼近*。在本章中我们将神经网络的设计看作是一个高维空间中的*曲线拟合(逼近)问题*，从而采用完全不同的方法进行设计。按照这种观点，学习等价于在多维空间中寻找一个能够最佳拟合训练数据的曲面，这里的“最佳拟合”准则是在某种统计意义上的最佳拟合。因此，泛化等价于利用这个多维曲面对测试数据进行插值。上述观点是径向基函数方法的出发点，径向基函数方法在某种程度上利用了多维空间中传统的严格插值法的研究成果。在神经网络的背景下，隐藏单元提供一个“函数”集，该函数集在输入模式(向量)扩展至隐藏空间时为其构建了一个任意的“基”；这个函数集中的函数就被称为*径向基函数*[1]。径向基函数首先是在实多变量插值问题的解中引入的。这方面的早期工作在 Powell(1985)中综述，而较新的工作则在 Light(1992b)中综述。径向基函数是目前数值分析研究中的一个主要领域。

最基本形式的*径向基函数(RBF)网络*的构成包括三层，其中每一层都有着完全不同的作用。输入层由一些源点(感知单元)组成，它们将网络与外界环境连结起来。第二层是网络中仅有的一个隐层，它的作用是从输入空间到隐藏空间之间进行非线性变换；在大多数情况下隐藏空间有较高的维数。输出层是线性的，它为作用于输入层的激活模式(信号)提供响应。关于非线性变换之后跟随线性变换的理论基础其数学依据可以追溯到 Cover(1965)的一篇早期论文。根据这篇文章，一个模式分类问题如果映射到一个高维空间将会比映射到一个低维空间更可能是线性可分的，这就是径向基函数网络的隐藏空间的维数通常都较高的原因。还有另外一个重要的原因，就是隐藏空间的维数与网络能否逼近一个光滑的输入－输出映射有着直接的联系(Mhaskar,1996;Niyogi and Girosi,1996)；隐藏空间的维数越高，逼近就越精确。

256

本章的组织

本章的主要部分组织如下。我们将有关构建 RBF 网络的基础放在 5.2 节和 5.4 节。分两个步骤来做到这一点。第一步，描述 Cover 关于模式可分的定理；将利用 XOR 问题来阐释该定理的应用。在 5.3 节将考虑插值问题及其他与 RBF 网络的关系。

在得到 RBF 网络如何工作的一个了解之后，我们将进入本章的第二部分，这部分包括 5.4 节至 5.9 节。在 5.4 节中讨论监督学习是一种不适定的超曲面重建问题的观点。在 5.5 节将详细论述 Tikhonov 的正则化理论及其在 RBF 网络中的应用。这个理论将很自然地导出在 5.6 节中正则化网络的公式。这类 RBF 网络对计算的要求很高。为了减少计算复杂性，在 5.7 节将讨论一个被称为广义 RBF 网络的改进正则化网络。在 5.8 节我们将重新讨论 XOR 问题，并且展示 RBF 网络是如何解决这个问题的。在 5.9 节将描述一种用于选择正则化参数恰当值的广义交叉确认方法，从而完成正则化理论的研究。

5.10节讨论RBF网络的逼近性质。5.11节将RBF网络与多层感知器模型作比较，这两种网络都是分层前馈网络的重要例子。

在5.12节讨论核回归估计，它是关于RBF网络的另一种观点的基础。我们将大量处理密度估计和核回归理论的统计学文献和RBF网络联系起来。

5.13节和5.14节是本章的最后一部分。在5.13节提出设计RBF网络的四个不同的学习策略。在5.14节描述一个用RBF网络进行模式分类的计算机试验。

在5.15节以某些关于RBF网络的最后的思想作为本章的结束。

5.2 模式可分性的Cover定理

当用径向基函数神经网络来解决一个复杂的模式分类任务时，问题的基本解决可以通过 257 用非线性方式将其变换到一个高维空间。它的潜在合理性来自模式可分性的Cover定理，该定理可以定性地表述如下(Cover,1965)：

将复杂的模式分类问题非线性地投射到高维空间将比投射到低维空间更可能是线性可分的。

从第3章对单层感知器的研究中知道，一旦模式具有线性可分性，则相应的分类问题相对而言就更容易解决。因此，我们通过研究模式的可分性可以深入了解RBF网络作为模式分类器是如何工作的。

考虑一族曲面，每一个曲面都自然地将输入空间自然地分成两个区域。用$\mathscr{X}$代表N个模式(向量)$\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$的集合，其中每一个模式都分属于两个类$\mathscr{X}_1$和$\mathscr{X}_2$中的一类。如果在这一族曲面中存在一个曲面能够将分别属于$\mathscr{X}_1$和$\mathscr{X}_2$的这些点分成两部分，我们就称这些点的二分(二元划分)关于这族曲面是可分的。对于每一个模式$\mathbf{x}\in\mathscr{X}$，定义一个由一组实值函数$\{\varphi_i(\mathbf{x})\mid i=1,2,\cdots,m_1\}$组成的向量，表示如下：

$$\boldsymbol{\varphi}(\mathbf{x})=[\varphi_1(\mathbf{x}),\varphi_2(\mathbf{x}),\cdots,\varphi_{m_1}(\mathbf{x})]^T \tag{5.1}$$

假设模式$\mathbf{x}$是m_0维输入空间的一个向量，则向量$\varphi(\mathbf{x})$将m_0维输入空间的点映射到新的m_1维空间的相应的点上。我们将$\varphi_i(\mathbf{x})$称为*隐藏函数*，因为它与前馈神经网络中的隐藏单元起着同样的作用。相应地，由隐藏函数集合$\{\varphi_i(\mathbf{x})\}_{i=1}^{m_1}$所生成的空间被称为*隐藏空间*或者*特征空间*。

我们称一个关于$\mathscr{X}$的二分$\{\mathscr{X}_1,\mathscr{X}_2\}$是$\varphi$可分的，如果存在一个$m_1$维的向量$\mathbf{w}$使得(Cover,1965)

$$\begin{aligned}\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x})>0,\quad &\mathbf{x}\in\mathscr{X}_1\\ \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x})<0,\quad &\mathbf{x}\in\mathscr{X}_2\end{aligned} \tag{5.2}$$

由方程

$$\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x})=0$$

定义的超平面描述φ空间(也就是隐藏空间)中的分离曲面。这个超平面的逆像，即

$$\mathbf{x}:\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x})=0 \tag{5.3}$$

定义输入空间中的分离曲面。

考虑一个利用r次模式向量坐标乘积的线性组合实现的一个自然类映射。与此种映射相

对应的分离曲面被称为 r 阶有理簇。一个 m_0 维空间的 r 阶有理簇可描述为输入向量 $\mathbf{x}$ 的坐标的一个 r 次齐次方程，表示为

$$\sum_{0 \leqslant i_1 \leqslant i_2 \leqslant \cdots \leqslant i_r \leqslant m_0} a_{i_1 i_2 \cdots i_r} x_{i_1} x_{i_2} \cdots x_{i_r} = 0 \tag{5.4}$$

其中 x_i 是输入向量 $\mathbf{x}$ 的第 i 个元素。为了用齐次形式来表达方程，将 x_0 的值置为单位值 1。 258 $\mathbf{x}$ 中项 x_i 的 r 阶乘积就是 $x_{i_1} x_{i_2} \cdots x_{i_r}$，被称为单项式。对于一个 m_0 维的输入空间在式(5.4)中一共有

$$\frac{(m_0 - r)!}{m_0 ! r!}$$

个单项式。式(5.4)所描述的分离曲面的类型的例子有超平面(一阶有理簇)、二次曲面(二阶有理簇)和超球面(带有某种线性限制系数的二次曲面)等。这些例子的说明见图 5-1，该图说明在二维输入空间中的五点的构形。通常情况下，线性可分性暗示着球面可分性，而球面可分性又暗示着二次可分性；然而反之不一定成立。

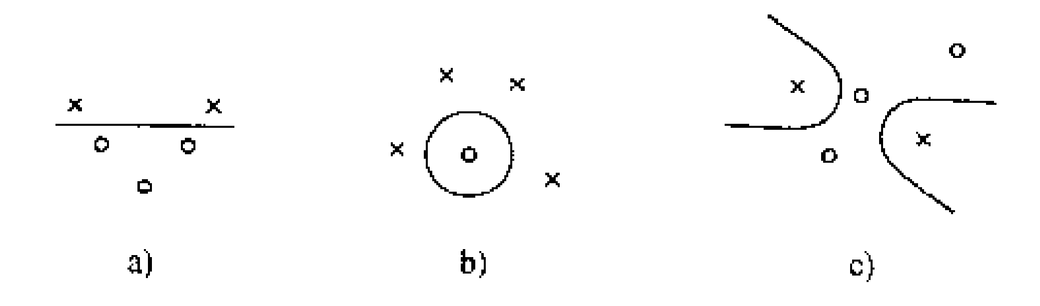


图 5-1 二维平面上的 5 个点的不同集合的 φ－可分的二分的 3 个例子：
a)线性可分的二分 b)球形可分的二分 c)二次可分的二分

在一个概率实验中，一个模式集合的可分性成为一个依赖于选择的二分以及输入空间中模式的分布的随机事件。假设激活模式 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$ 是根据输入空间中的概率特性而独立选取的。同时假设所有的关于$\mathscr{X} = \{\mathbf{x}_i\}_{i=1}^N$的二分都是等可能的。令 $P(N, m_1)$表示某一随机选取的二分是 φ 可分的概率，这里被选中的分离曲面的类具有 m_1 维的自由度。根据 Cover (1965)，我们可以将 $P(N, m_1)$表述为

$$P(N, m_1) = \left(\frac{1}{2}\right)^{N-1} \sum_{m=0}^{m_1 - 1} \binom{N-1}{m} \tag{5.5}$$

这里，包括 $N-1$ 和 m 的二项式系数定义如下：

$$\binom{l}{m} = \frac{l(l-1)\cdots(l-m+1)}{m!}$$

259

式(5.5) 体现 Cover 的可分性定理对于随机模式[2]的本质。它说明累计二项概率分布，相当于抛$(N-1)$次硬币有$(m_1 - 1)$次或更少次头像向上的概率。

尽管在式(5.5)的推导中遇见的隐藏单元曲面是一个多项式的形式，从而与我们通常在径向基函数网络中用到的有所不同，但是该式的核心内容却具有普遍的适用性。特别地，若隐藏空间的维数 m_1 越高，则概率 $P(N, m_1)$就越趋向于 1。总之，关于模式可分性的 Cover 定理主要包含下面两个基本部分：

1. 由 $\varphi_i(\mathbf{x})$定义的隐藏函数的非线性构成，这里 $\mathbf{x}$ 是输入向量，且 $i = 1, 2, \cdots, m_1$。

2. 高维数的隐藏空间，这里的高维数是相对于输入空间而言的。维数由赋给 m_1 的值

(即隐藏单元的个数)决定。

如前所述，通常将一个复杂的模式分类问题非线性地投射到高维数空间将会比投射到低维数空间更可能是线性可分的。但是需要强调的是，有时使用非线性映射(即第1部分)就足够导致线性可分，而且不必升高隐藏单元空间维数，如下面例子所说明的那样。

例 5.1　XOR 问题　为了说明模式的 φ 可分性思想的意义，考虑一个简单却又十分重要的 XOR 问题。在 XOR 问题中有四个二维输入空间上的点(模式)：(1,1)，(0,1)，(0,0)和(1,0)，如图 5-2a。要求建立一个模式分类器产生二值输出响应，其中点(1,1)或(0,0)对应于输出 0，点(1,0)或(0,1)对应于输出 1。因此在输入空间中依 Hamming 距离最近的点映射到在输出空间中最大分离的区域。

定义一对 Gauss 隐藏函数如下：

$$\varphi_1(\mathbf{x}) = e^{-\|\mathbf{x}-\mathbf{t}_1\|^2}, \qquad \mathbf{t}_1 = [1,1]^T$$

$$\varphi_2(\mathbf{x}) = e^{-\|\mathbf{x}-\mathbf{t}_2\|^2}, \qquad \mathbf{t}_2 = [0,0]^T$$

这样我们可以得到以上四个点作为输入时的结果，如表 5-1 所示。如图 5-2b，输入模式被映射到 $\varphi_1 - \varphi_2$ 平面上。这里我们可以看到输入(0,1)，(1,0)与剩下的两个输入(1,1)，(0,0)是线性可分的。然后，我们将 $\varphi_1(\mathbf{x})$ 和 $\varphi_2(\mathbf{x})$ 作为一个线性分类器如感知器模型的输入，则 XOR 问题就迎刃而解了。■

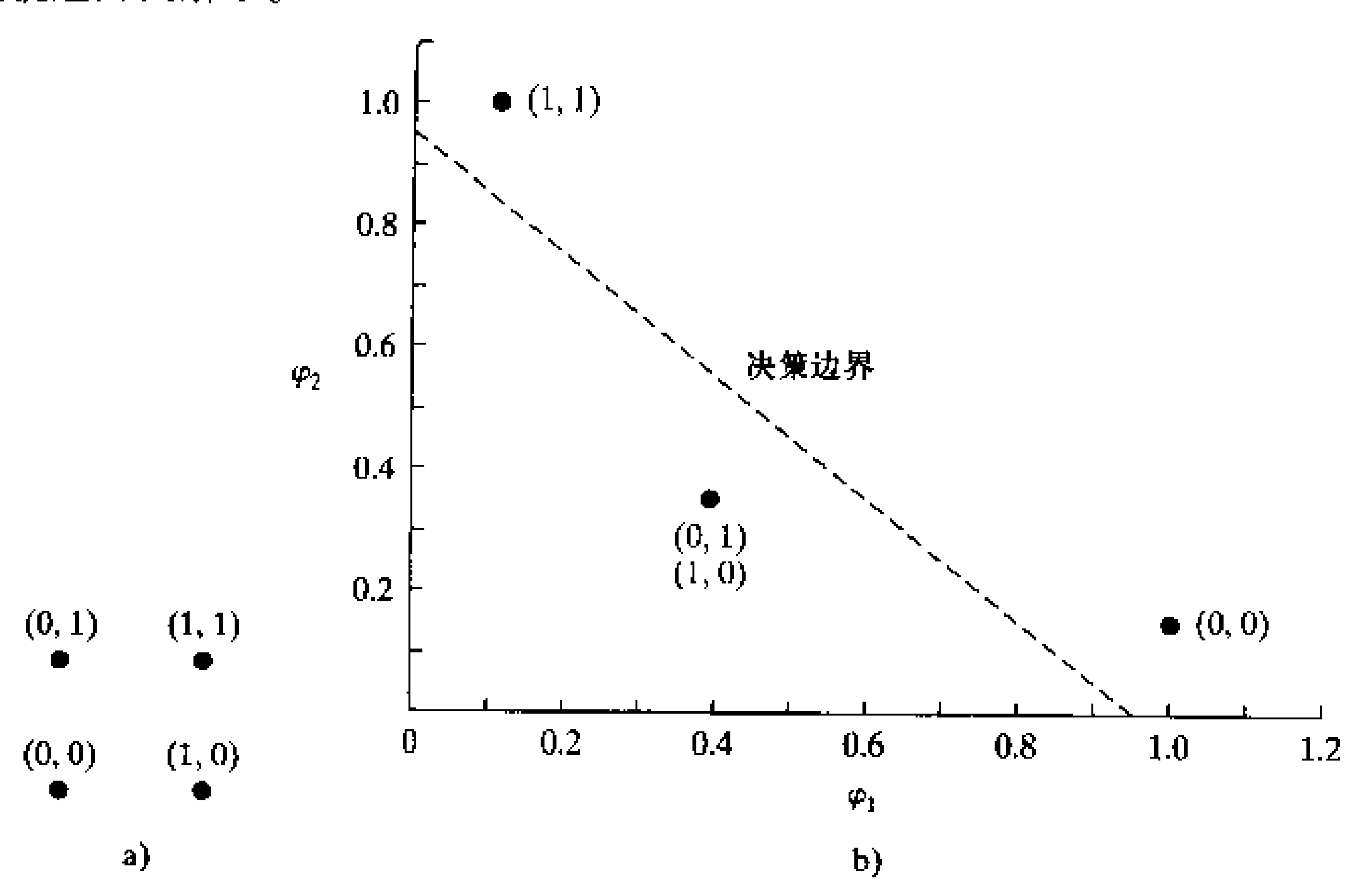


图 5-2

a)XOR 问题的 4 个模式　b)决策图

表 5-1　用于例 5.1 的 XOR 问题的隐藏函数设置

输入模式 x	第一隐藏函数 $\varphi_1(\mathbf{x})$	第二隐藏函数 $\varphi_2(\mathbf{x})$
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(0,0)	0.1353	1
(1,0)	0.3678	0.3678

在这个例子中隐藏空间的维数相对于输入空间并没有增加。也就是说，以 Gauss 函数作为非线性的隐藏函数，足以将 XOR 问题转化为一个线性可分问题。 260

曲面的分离能力

式(5.5)对于在多维空间中随机指定输入模式线性可分的期望最大数目有重要意义。为了研究这个问题，如前所述将 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$ 视为一个随机模式(向量)序列。令 N 为一个随机变量，定义为该序列为 φ 可分时的最大整数，这里 φ 具有 m_1 的自由度。于是由式(5.5)我们可以导出当 $N=n$ 时的概率

$$\text{Prob}(N=n)=P(n,m_1)-P(n+1,m_1)=\left(\frac{1}{2}\right)^n\binom{n-1}{m_1-1},n=0,1,2,\cdots \quad (5.6)$$

为了解释上述结果，我们回想一下负二项分布的定义。该分布相当于在一组重复的 Bernoulli 实验中有 r 次成功、k 次失败且最后一次是成功的概率。在这种概率实验中，每一次实验只有两种结果，不是成功就是失败，并且成功和失败的概率在整组实验中都是相同的。令 p 代表成功的概率，q 代表失败的概率，$p+q=1$。负二项分布定义(Feller, 1968)如下： 261

$$f(k;r,p)=p^rq^k\binom{r+k-1}{k}$$

在 $p=q=1/2$(即成功和失败具有相等的概率)且 $k+r=n$ 的特殊情况下，上述的负二项分布将变为

$$f\left(k;n-k,\frac{1}{2}\right)=\left(\frac{1}{2}\right)^n\binom{n-1}{k},n=0,1,2,\cdots$$

根据上述定义，我们现在可以看出由式(5.6)所表示的结果正是负二项分布，只不过右移了 m_1 个单位且具有参数 m_1 和 1/2。这样，N 相当于在一组抛硬币的实验中出现第 m_1 次失败的“等待时间”。随机变量 N 的期望和中位数分别为

$$E[N]=2m_1 \quad (5.7)$$

和

$$\text{Median}[N]=2m_1 \quad (5.8)$$

因此，我们可以得到 Cover 定理的一个推论，用著名的渐近结果的形式可表述如下：

一组随机指定的输入模式(向量)的集合在 m_1 维空间中线性可分，它的元素数目的最大期望等于 $2m_1$。

该结果表明，$2m_1$ 是对一族具有 m_1 维自由度的决策曲面的分离能力的自然定义。在一定程度上，一个曲面的分离能力与第 2 章讨论的 VC 维数的概念有着紧密的联系。

5.3 插值问题

从关于模式可分性的 Cover 定理得到的重要思想是在解决一个非线性可分的模式分类问题时，如果将输入空间映射到一个新的维数足够高的空间去，将会有助于问题的解决。基本说来用一个非线性变换将一个非线性可分的分类问题转变为一个线性可分问题。同样地，我们可以用非线性变换将一个复杂的非线性滤波问题转化为一个较简单的线性滤波问题。

现在考虑一个由输入层、一个中间层和只有一个输出单元的输出层组成的前馈网络。我们选择只有一个输出单元的输出层的目的主要是为了简化说明又不失一般性。设计这个网络 262 实现从输入空间到隐藏空间的一个非线性映射，随后从隐藏空间到输出空间则是线性映射。令 m_0 为输入空间的维数。这样从总体上看这个网络就相当于一个从 m_0 维输入空间到一维输出空间的映射，可以写成如下形式：

$$s: \mathbb{R}^{m_0} \rightarrow \mathbb{R}^1 \tag{5.9}$$

我们可以将映射 s 视为一个*超曲面*(图)$\Gamma \subset \mathbb{R}^{m_0+1}$，就好像我们可以将一个最基本的映射 s: $\mathbb{R}^1 \rightarrow \mathbb{R}^1$，其中 $s(x)=x^2$，视为 $\mathbb{R}^2$ 空间中的一条抛物线一样。超曲面 Γ 作为输入的函数是输出空间的多维曲面。在实际情况下，曲面 Γ 是未知的，并且训练数据中通常带有噪声。学习中的训练阶段和泛化阶段可叙述如下：

- 训练阶段由曲面 Γ 的拟合过程的最优化构成，它根据以输入－输出样本(模式)形式呈现给网络的已知数据进行。
- 泛化阶段的任务就是在数据点之间进行插值，插值是在真实曲面 Γ 的最佳逼近的拟合过程产生的约束曲面上进行的。

这样我们将引出具有悠久历史的高维空间多变量插值理论(Davis, 1963)。从严格意义上说，插值问题可以叙述如下：

给定一个包含 N 个不同点的集合 $\{\mathbf{x}_i \in \mathbb{R}^{m_0} \mid i=1,2,\cdots,N\}$ 和相应的 N 个实数的一个集合 $\{d_i \in \mathbb{R}^1 \mid i=1,2,\cdots,N\}$，寻找一个函数 F: $\mathbb{R}^N \rightarrow \mathbb{R}^1$ 满足下述插值条件：

$$F(\mathbf{x}_i) = d_i, \qquad i = 1,2,\cdots,N \tag{5.10}$$

对于这里所述的严格插值来说，插值曲面(即函数 F)必须通过所有的训练数据点。

径向基函数(RBF)技术就是要选择一个函数 F 具有下列形式(Powell, 1988)：

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i \varphi(\|\mathbf{x}-\mathbf{x}_i\|) \tag{5.11}$$

其中 $\{\varphi(\|\mathbf{x}-\mathbf{x}_i\|) \mid i=1,2,\cdots,N\}$ 是 N 个任意(一般地是线性)函数的集合，称为*径向基函数*；$\|\cdot\|$ 表示范数，通常是欧儿里德范数。已知数据 $\mathbf{x}_i \in \mathbb{R}^{m_0}$，$i=1, 2, \cdots, N$ 是径向基函数的中心。

将(5.10)的插值条件代入式(5.11)中，我们可以得到一组关于未知系数(权值)的展开 $\{w_i\}$ 的线性方程组：

263

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix} \tag{5.12}$$

其中 $$\varphi_{ji} = \varphi(\|\mathbf{x}_j - \mathbf{x}_i\|),\ (j,\ i) = 1,\ 2,\ \cdots,\ N \tag{5.13}$$

令 $$\mathbf{d} = [d_1,\ d_2,\ \cdots,\ d_N]^T,\ \mathbf{w} = [w_1,\ w_2,\ \cdots,\ w_N]^T$$

上式中的 $N \times 1$ 向量 $\mathbf{d}$ 和 $\mathbf{w}$ 分别表示期望输出向量和连结权值向量，其中 N 表示训练样本的长度。令 $\mathbf{\Phi}$ 表示元素为 φ_{ji} 的 $N \times N$ 阶的矩阵：

$$\mathbf{\Phi} = \{\varphi_{ji} \mid (j,i) = 1,2,\cdots,N\} \tag{5.14}$$

我们称该矩阵为插值矩阵。于是式(5.12)可以写成紧凑形式

$$\mathbf{\Phi w} = \mathbf{x} \tag{5.15}$$

假设 $\mathbf{\Phi}$ 为非奇异矩阵，因此而存在 $\mathbf{\Phi}^{-1}$。这样我们就可以从式(5.15)中解出权值向量 $\mathbf{w}$，表示为

$$\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{x} \tag{5.16}$$

问题的关键是：我们怎么能保证插值矩阵 $\mathbf{\Phi}$ 是非奇异的？可以证明，对于大量径向基函数来说在某种条件下上述问题的答案可以由下面的重要定理给出。

Micchelli 定理

Micchelli(1986)证明了如下定理：

如果 $\{\mathbf{x}_i\}_{i=1}^{N}$ 是 $\mathbb{R}^{m_0}$ 中 N 个互不相同的点的集合，则 $N\times N$ 阶的插值矩阵 $\mathbf{\Phi}$(第 ji 个元素是 $\varphi_{ji}=\varphi(\|\mathbf{x}_j-\mathbf{x}_i\|)$)是非奇异的。

有大量的径向基函数满足 Micchelli 定理，包括下面三个在径向基函数网络中有重要地位的函数：

1. 多二次(Multiquadrics)函数：

$$\varphi(r) = (r^2 + c^2)^{1/2} \qquad c > 0, r \in \mathbb{R} \tag{5.17}$$

2. 逆多二次(Inverse multiquadrics)函数：

$$\varphi(r) = \frac{1}{(r^2 + c^2)^{1/2}} \qquad c > 0, r \in \mathbb{R} \tag{5.18}$$

3. Gauss 函数：

264

$$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \qquad \sigma > 0, r \in \mathbb{R} \tag{5.19}$$

多二次函数和逆多二次函数都应归功于 Hardy(1971)。

为了使式(5.17)至(5.19)所示的径向基函数是非奇异的，必须使所有的输入点 $\{\mathbf{x}_i\}_{i=1}^{N}$ 互不相同。这就是使插值矩阵 $\mathbf{\Phi}$ 非奇异的全部要求，与所给样本的长度 N 和向量(点)$\mathbf{x}_i$ 的维数 m_0 无关。

式(5.18)的逆多二次函数和式(5.19)的 Gauss 函数具有一个共同的性质：它们都是局部化的函数，因为当 $r\to\infty$ 时，$\varphi(r)\to 0$。以上面两个函数作为径向基函数所组成的插值矩阵 $\mathbf{\Phi}$ 都是正定的。与此相反，而由式(5.17)所定义的多二次函数是非局部性函数，因为当 $r\to\infty$ 时，$\varphi(r)$ 是无界的；与其相对应的插值矩阵 $\mathbf{\Phi}$ 有 $(N-1)$ 个负的特征值，只有一个正的特征值，所以不是正定的(Micchelli 1986)。但值得注意的是在 Hardy 的多二次函数基础上建立的插值矩阵 $\mathbf{\Phi}$ 却是非奇异的，因此适合在 RBF 网络设计中应用。

一个更加值得注意的是径向基函数若是无限增长的，例如多二次函数，与其他产生正定插值矩阵的函数相比，它能以更高的精度逼近一个光滑的输入－输出映射。Powell(1988)讨论这个令人惊奇的结果。

5.4 作为不适定超曲面重建问题的监督学习

在某些任务中由于对新数据具有较差的泛化性能，这样利用上述严格的插值方法来训练

一个 RBF 网络并不是一个好办法。这是因为如果训练样本中的数据点的数目远远大于固有的物理过程的自由度，并且我们限制径向基函数的个数与数据点的个数是相同的，这样问题就为超定的。结果神经网络就会因为输入数据的特性(idiosyncrasy)或者噪声干扰而拟合到一个错误的曲面，从而导致泛化性能降低(Broomhead and Lowe，1988)。

为了进一步加深对过拟合问题的理解并且如何克服这个问题，我们可以先回到这样观点：训练神经网络使其能够根据输入模式找到相应的输出模式，它的设计相当于学习一个超曲面(即多维映射)使其能够根据输入确定输出。换句话说，*学习可以被视为给定一组可能是稀疏的数据点的超曲面重建问题*。

根据 Keller(1976)和 Kirsch(1996)，如果有相关两个问题，系统地解决其中的任意一个问题都必须部分地或者全部地知道关于另一个问题的知识，那么我们就称这两个问题是互逆的。通常我们发现其中一个问题比另一个问题研究得早，并且可能研究得更透彻，那么这个问题就被称为正问题(direct problem)，而另一个问题就被称为逆问题(inverse problem)。然而从数学角度来说，正问题和逆问题之间有着更重要的区别。特别地，所研究问题是适定的(well-posed)还是不适定的(ill-posed)。“适定”这个术语在20世纪初从 Hadamard 的那个时期起就已经在应用数学中使用。为了解释这个术语，假设我们在度量空间有一个定义域 X 和一个值域 Y，它们由一个固定的但是未知的映射 f 联系着。如果下面三个条件均满足的话，我们就称映射 f 的重建问题是*适定的*(Tikhonov and Arsenin，1977；Morozov，1993；Kirsch，1996)：

1. *存在性*。对每一个输入向量 $\mathbf{x}\in\mathscr{X}$，都存在一个输出 $y=f(\mathbf{x})$，其中 $y\in\mathscr{Y}$。

2. *惟一性*。对任何一对输入向量 $\mathbf{x}$，$\mathbf{t}\in\mathscr{X}$，当且仅当 $\mathbf{x}=\mathbf{t}$ 时有 $f(\mathbf{x})=f(\mathbf{t})$。

3. *连续性*。映射是连续的，即对任何 $\varepsilon>0$，存在 $\delta=\delta(\varepsilon)$ 使得当 $\rho_x(\mathbf{x},\mathbf{t})<\delta$ 时，$\rho_y(f(\mathbf{x}),f(\mathbf{t}))<\varepsilon$ 成立。其中 $\rho(\cdot,\cdot)$ 表示两个变量在其所属空间中的距离。这一准则如图5-3所示。连续性通常也被称为*稳定性*。

如果上述的任何一项条件不满足，那么问题就称为*不适定的*。从根本上说，一个问题如果是不适定的，说明大量的数据集合里只包含着很少一部分的有用信息。

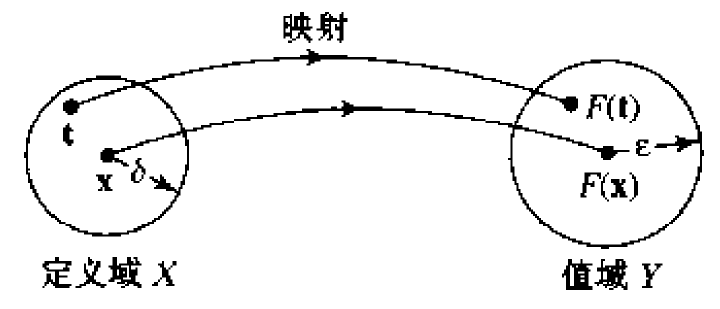


图5-3 定义域$\mathscr{X}$(输入)到值域$\mathscr{Y}$的映射示例

在我们现时的背景下，负责产生训练数据(例如语音、图象、雷达信号、声纳信号和地震数据等)的物理现象是适定的正问题。然而，从这些数据的物理形式学习，看作超曲面的重建问题，基于后面的原因却是一个不适定的逆问题。原因如下：第一，存在性准则可能不满足，因为对于每一个输入来说，其不同的输出并不一定存在。第二，训练样本中可能没有完整重建输入-输出映射所需的足够信息，因而惟一性准则可能不满足。第三，现实生活中训练数据不可避免出现噪声以及不精确性，增加了输入-输出映射重建的不确定性。特别地，若输入中所含的噪声水平太高，对于定义域$\mathscr{X}$的特定输入 $\mathbf{x}$，由神经网络所产生的输出结果可能超出值域$\mathscr{Y}$的范围；换句话说，连续性准则可能不满足。如果一个学习问题不具有连续性，那么计算所得的输入-输出映射将和学习问题的真解毫无关系。除非预先知道一些关于输入-输出映射的先验信息，否则这个问题是不可克服的。在这个背景下，我们引用 Lanczos 关于线性微分算子所作陈述是恰当的：“信息的缺乏并不能靠任何数学技巧来弥补。”

我们将在下一节讨论如何通过正则化方法将一个不适定问题转变成一个适定问题[3]。

5.5 正则化理论

1963 年 Tikhonov 提出了一种新的方法用以解决不适定问题[4]，该方法就是正则化方法。在曲面重建的问题上，正则化的基本思想就是通过某些含有解的先验知识的非负的辅助泛函来使解稳定。先验知识的一般形式涉及假设输入 - 输出映射函数(即重建问题的解)是光滑的，意味着相似的输入对应着相似的输出。

进一步，我们将用于逼近的输入 - 输出数据(即训练样本)集合描述如下：

$$\begin{aligned}&\text{输入信号}:\mathbf{x}_i \in \mathbb{R}^{m_0}, \quad i = 1,2,\cdots,N\\&\text{期望响应}:d_i \in \mathbb{R}^1, \quad i = 1,2,\cdots,N\end{aligned} \tag{5.20}$$

注意这里假定输出是一维的。这种假设并不会限制这里讨论的正则化理论的一般性应用。用 $F(\mathbf{x})$表示逼近函数，这里为了方便表达，我们在变量中省掉了神经网络的权值向量 $\mathbf{w}$。从根本上说，Tikhonov 的正则化理论包含两项：

1. *标准误差项*。该项用$\mathscr{E}_s(F)$表示，用以度量对于训练样本 $i=1, 2, \cdots, N$ 的期望(目标)响应 d_i 和实际响应 y_i 之间的标准误差(距离)。具体定义为

$$\mathscr{E}_s(F) = \frac{1}{2}\sum_{i=1}^{N}(d_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]^2 \tag{5.21}$$

其中，我们引入比例因子 1/2 是为了与前面几章保持一致。

2. *正则化项*。第 2 项用$\mathscr{E}_c(F)$表示，依赖于逼近函数 $F(\mathbf{x})$的“几何”性质。具体定义为

$$\mathscr{E}_c(F) = \frac{1}{2}\|\mathbf{D}F\|^2 \tag{5.22}$$

其中，$\mathbf{D}$ 是线性微分算子。关于解(即输入 - 输出映射 $F(\mathbf{x})$)的形式的先验知识就包含在算子 $\mathbf{D}$ 中，这就自然使得 $\mathbf{D}$ 的选取与所解的问题有关。我们也称 $\mathbf{D}$ 为稳定因子(stabilizer)，因为它使正则化问题的解稳定，使解光滑从而满足连续性的要求。但是，光滑性意味着连续性，而相反未必为真。

用于处理式(5.22)所描述情况的解析方法是建立在函数空间[5]的概念之上的。函数空间指的是函数的*赋范空间*[6]。在这样的多维(严格说来是无限多维)空间中，一个连续函数由一个向量来表示。在这种几何图像意义上，我们就可以在线性微分算子和矩阵之间建立深刻的联系。由此对线性系统的分析就可以转变为对线性微分方程的分析(Lanczos, 1964)。 267

于是，式(5.22)中的符号 $\|\cdot\|$ 表示定义在 $\mathbf{D}F(\mathbf{x})$所属空间上的范数。一般情况下这里所使用的函数空间指的是包含了所有实值函数 $f(\mathbf{x})$，$\mathbf{x}\in\mathbb{R}^{m_0}$ 的 L_2 空间，其中 $\|f(\mathbf{x})\|^2$ 是 Lebesgue 可积的。这里用函数 $f(\mathbf{x})$表示实际定义的负责产生输入 - 输出数据对$\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$的物理过程。更多细节参见注释[7]。

正则化理论要求最小化的量为

$$\mathscr{E}(F) = \mathscr{E}_s(F) + \lambda\mathscr{E}_c(F) = \frac{1}{2}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]^2 + \frac{1}{2}\lambda\|\mathbf{D}F\|^2 \tag{5.23}$$

其中 λ 是一个正的实数，叫做*正则化参数*；$\mathscr{E}(F)$叫做 Tikhonov *泛函*。一个泛函映射函数(定义在某个适当的函数空间)到实直线。使 Tikhonov 泛函$\mathscr{E}(F)$最小的解函数(也就是正则化问题的解)记为 $F_\lambda(\mathbf{x})$。

在某种意义上，我们可以将正则化参数 λ 视为一个指示器，用来指示所给的数据集作为确定解 $F_\lambda(\mathbf{x})$ 的样本的充分性。特别在极限情况下，当 $\lambda \to 0$ 时，表明该问题不受约束，问题解 $F_\lambda(\mathbf{x})$ 完全决定于所给样本。另一方面，当 $\lambda \to \infty$ 时，表明仅由算子 $\mathbf{D}$ 所定义的先验光滑条件就足以得到问题的解 $F_\lambda(\mathbf{x})$，这也是所给样本完全不可信的另一种说法。在实际应用中，正则化参数 λ 取值在上述两个极限值之间，使得样本数据和先验信息都对解 $F_\lambda(\mathbf{x})$ 作了贡献。因此正则化项 $\mathscr{E}_c(F)$ 表示一个*模型复杂性－惩罚函数*，其对最终解的影响取决于正则化参数 λ 的大小。

另外可将正则化看作提供第2章讨论的偏置－方差困境的一个可行的解。具体地，在正则化参数 λ 的最优选择的设计中通过融合恰当的先验知识使得学习问题的解在模型偏置和模型方差之间达到一个满意的平衡。

Tikhonov 泛函的 Fréchet 微分

*正则化原理*可以叙述如下：

求使 Tikhonov 泛函 $\mathscr{E}(F)$ 最小的函数 $F_\lambda(\mathbf{x})$，其中，Tikhonov 泛函由

$$\mathscr{E}(F) = \mathscr{E}_s(F) + \lambda \mathscr{E}_c(F)$$

定义，其中 $\mathscr{E}_s(F)$ 是标准误差项，$\mathscr{E}_c(F)$ 是正则化项，而 λ 是正则化参数。

[268] 为进行最小化代价泛函 $\mathscr{E}(F)$，我们首先需要求 $\mathscr{E}(F)$ 微分的规则。我们可以用 Fréchet 微分来处理这件事。在初等微积分中，曲线上某点的切线是在该点邻域上的曲线的最佳逼近直线。同理，一个泛函的 Fréchet 微分可以解释为一个最佳局部线性逼近。这样泛函 $\mathscr{E}(F)$ 的 Fréchet 微分可正式定义如下(Dorny, 1975; Debnath and Mikusiński, 1990; de Figueiredo and Chen, 1993)：

$$d\mathscr{E}(F,h) = \left[\frac{d}{d\beta}\mathscr{E}(F+\beta h)\right]_{\beta=0} \tag{5.24}$$

上式中 $h(\mathbf{x})$ 是一个固定的关于向量 $\mathbf{x}$ 的函数。在式(5.24)中应用通常的微分法则。函数 $F(\mathbf{x})$ 为泛函 $\mathscr{E}(F)$ 的一个相对极值的必要条件是对于所有的 $h \in \mathscr{H}$，泛函 $\mathscr{E}(F)$ 的 Fréchet 微分 $d\mathscr{E}(F,h)$ 在 $F(\mathbf{x})$ 处均为零，表示为

$$d\mathscr{E}(F,h) = d\mathscr{E}_s(F,h) + \lambda d\mathscr{E}_c(F,h) = 0 \tag{5.25}$$

其中 $d\mathscr{E}_s(F,h)$ 和 $d\mathscr{E}_c(F,h)$ 分别是泛函 $\mathscr{E}_s(F)$ 和 $\mathscr{E}_c(F)$ 的 Fréchet 微分。

计算式(5.21)标准误差项 $\mathscr{E}_s(F, h)$ 的 Fréchet 微分如下：

$$\begin{aligned} d\mathscr{E}_s(F,h) &= \left[\frac{d}{d\beta}\mathscr{E}_s(F+\beta h)\right]_{\beta=0} = \left[\frac{1}{2}\frac{d}{d\beta}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)]^2\right]_{\beta=0} \\ &= -\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)]h(\mathbf{x}_i)\Big|_{\beta=0} = -\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]h(\mathbf{x}_i) \end{aligned} \tag{5.26}$$

在讨论的这一点上，我们发现引入 *Riesz 表示定理*是有益的(Debnath and Mikusiński, 1990; Kirsch, 1996)，它可陈述如下：

令 f 为 Hilbert 空间(即一个完备的内积空间[8]，用符号 $\mathscr{H}$ 表示)上的一个有界线性泛函。

存在一个 $h_0 \in \mathscr{H}$，使得对所有 $h \in \mathscr{H}$ 都有

$$f = (h, h_0)_{\mathscr{H}}$$

且 $\|f\|_{\tilde{\mathscr{H}}} = \|h_0\|_{\mathscr{H}}$ 其中 $\tilde{\mathscr{H}}$ 是 Hilbert 空间 $\mathscr{H}$ 的对偶空间或者共轭空间。

这里所用的符号 $(\cdot,\cdot)_{\mathscr{H}}$ 表示 $\mathscr{H}$ 空间上两个函数的内积(纯量积)。因此，根据 Riesz 表示定理，我们可以重写式(5.26)的 Fréchet 微分 $d\mathscr{E}_s(F,h)$ 如下：

$$d\mathscr{E}_s(F,h) = -\left(h, \sum_{i=1}^{N}(d_i - F)\delta_{\mathbf{x}_i}\right)_{\mathscr{H}} \tag{5.27}$$

式中 $\delta_{\mathbf{x}_i}$ 表示以 $\mathbf{x}_i$ 为中心的 $\mathbf{x}$ 的 Dirac delta 分布；即

$$\delta_{\mathbf{x}_i}(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_i) \tag{5.28}$$

269

下面计算式(5.22)的正则化项 $\mathscr{E}_c(F)$ 的 Fréchet 微分。用上面同样的方法我们可以得到

$$\begin{aligned} d\mathscr{E}_c(F,h) &= \frac{d}{d\beta}\mathscr{E}_c(F+\beta h)\Big|_{\beta=0} = \frac{1}{2}\frac{d}{d\beta}\int_{\mathbb{R}^{m_0}}(\mathbf{D}[F+\beta h])^2 d\mathbf{x}\Big|_{\beta=0} \\ &= \int_{\mathbb{R}^{m_0}}\mathbf{D}[F+\beta h]\mathbf{D}h d\mathbf{x}\Big|_{\beta=0} = \int_{\mathbb{R}^{m_0}}\mathbf{D}F\mathbf{D}h d\mathbf{x} = (\mathbf{D}h, \mathbf{D}F)_{\mathscr{H}} \end{aligned} \tag{5.29}$$

其中 $(\mathbf{D}h, \mathbf{D}F)_{\mathscr{H}}$ 是函数 $\mathbf{D}h(\mathbf{x})$ 和 $\mathbf{D}F(\mathbf{x})$ 的内积，函数 $\mathbf{D}h(\mathbf{x})$ 和 $\mathbf{D}F(\mathbf{x})$ 分别代表了微分算子 $\mathbf{D}$ 作用在 $h(\mathbf{x})$ 和 $F(\mathbf{x})$ 上的结果。

Euler-Lagrange 方程

给定一个线性微分算子 $\mathbf{D}$，我们可以惟一确定它的伴随算子 $\tilde{\mathbf{D}}$，使得对任一对足够可微且满足恰当的边界条件的函数 $u(\mathbf{x})$ 和 $v(\mathbf{x})$ 有

$$\int_{\mathbb{R}^m} u(\mathbf{x})\mathbf{D}v(\mathbf{x})d\mathbf{x} = \int_{\mathbb{R}^m} v(\mathbf{x})\tilde{\mathbf{D}}u(\mathbf{x})d\mathbf{x} \tag{5.30}$$

等式(5.30)叫做 Green 恒等式；它为通过给定微分算子 $\mathbf{D}$ 来确定其伴随算子 $\tilde{\mathbf{D}}$ 提供一个数学基础。将 $\mathbf{D}$ 看作一个矩阵，则其伴随算子 $\tilde{\mathbf{D}}$ 的作用类似于一个转置矩阵的作用。

比较式(5.30)的左边和式(5.29)的第四行，我们可作出如下恒等式：

$$u(\mathbf{x}) = \mathbf{D}F(\mathbf{x}), \mathbf{D}v(\mathbf{x}) = \mathbf{D}h(\mathbf{x})$$

根据 Green 恒等式可将式(5.29)重写为等价形式

$$d\mathscr{E}_c(F,h) = \int_{\mathbb{R}^{m_0}} h(\mathbf{x})\tilde{\mathbf{D}}\mathbf{D}F(\mathbf{x})d\mathbf{x} = (h, \tilde{\mathbf{D}}\mathbf{D}F)_{\mathscr{H}} \tag{5.31}$$

其中 $\tilde{\mathbf{D}}$ 是 $\mathbf{D}$ 的伴随算子。

将式(5.27)和(5.31)代入极值条件(5.25)中，可以重新得到 Fréchet 微分 $d\mathscr{E}(F,h)$ 如下：

$$d\mathscr{E}(F,h) = \left(h, \left[\tilde{\mathbf{D}}\mathbf{D}F - \frac{1}{\lambda}\sum_{i=1}^{N}(d_i - F)\delta_{\mathbf{x}_i}\right]\right)_{\mathscr{H}} \tag{5.32}$$

270

因为正则化参数 λ 通常取开区间 $(0,\infty)$ 上的某个值，所以当且仅当下列条件在广义函数意义下满足时，对于 $\mathscr{H}$ 空间中的所有函数 $h(\mathbf{x})$，Fréchet 微分 $d\mathscr{E}(F,h)$ 才为零：

$$\tilde{\mathbf{D}}\mathbf{D}F_\lambda - \frac{1}{\lambda}\sum_{i=1}^{N}(d_i - F)\delta_{\mathbf{x}_i} = 0$$

或者等价地，

$$\tilde{\mathbf{D}}\mathbf{D}F_\lambda(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\delta(\mathbf{x} - \mathbf{x}_i) \tag{5.33}$$

式(5.33)是 Tikhonov 泛函$\mathscr{E}(F)$的 Euler-Lagrange 方程；它定义 Tikhonov 泛函$\mathscr{E}(F)$在 $F_\lambda(\mathbf{x})$处有极值的必要条件(Debnath and Mikusiński, 1990)。

Green 函数

式(5.33)表示逼近函数 F 的偏微分方程。该方程的解是由方程右边的积分变换组成的。

令 $G(\mathbf{x},\boldsymbol{\xi})$表示向量 $\mathbf{x}$ 和 $\boldsymbol{\xi}$ 的一个函数，两个向量的地位相同，但它们的目的不同：向量 $\mathbf{x}$ 作为参数，而向量 $\boldsymbol{\xi}$ 则作为自变量。对于给定的线性微分算子 $\mathbf{L}$，我们规定函数 $G(\mathbf{x},\boldsymbol{\xi})$满足如下条件(Courant and Hilbert, 1970)：

1. 对于固定的 $\boldsymbol{\xi}$，$G(\mathbf{x},\boldsymbol{\xi})$是 $\mathbf{x}$ 的函数，且满足规定的边界条件。

2. 除了在点 $\mathbf{x}=\boldsymbol{\xi}$ 外，$G(\mathbf{x},\boldsymbol{\xi})$对于 $\mathbf{x}$ 的导数是连续的。导数的次数由线性算子 $\mathbf{L}$ 的阶数决定。

3. 将 $G(\mathbf{x},\boldsymbol{\xi})$看作 $\mathbf{x}$ 的函数，除了在点 $\mathbf{x}=\boldsymbol{\xi}$ 奇异外，它满足偏微分方程

$$\mathbf{L}G(\mathbf{x},\boldsymbol{\xi}) = 0 \tag{5.34}$$

也即函数 $G(\mathbf{x},\boldsymbol{\xi})$满足(在广义函数的意义下)

$$\mathbf{L}G(\mathbf{x},\boldsymbol{\xi}) = \delta(\mathbf{x} - \boldsymbol{\xi}) \tag{5.35}$$

其中，$\delta(\mathbf{x}-\boldsymbol{\xi})$如前定义是位于点 $\mathbf{x}=\boldsymbol{\xi}$ 的 Dirac delta 函数。

因此上述的函数 $G(\mathbf{x},\boldsymbol{\xi})$叫做微分算子 $\mathbf{L}$ 的 Green 函数。Green 函数对于线性微分算子的作用类似于一个矩阵的逆矩阵对该矩阵方程的作用。

令 $\varphi(\mathbf{x})$表示一个关于 $\mathbf{x}\in\mathbb{R}^{m_0}$ 的连续或者分段连续的函数。那么函数

271

$$F(\mathbf{x}) = \int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\,d\boldsymbol{\xi} \tag{5.36}$$

就是微分方程

$$\mathbf{L}F(\mathbf{x}) = \varphi(\mathbf{x}) \tag{5.37}$$

的解，其中 $G(\mathbf{x},\boldsymbol{\xi})$是线性微分算子 $\mathbf{L}$ 的 Green 函数(Courant and Hilbert, 1970)。

为了证明 $F(\mathbf{x})$为(5.37)的解，我们将微分算子 $\mathbf{L}$ 作用于式(5.36)的两端，可得

$$\mathbf{L}F(\mathbf{x}) = \mathbf{L}\int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\,d(\boldsymbol{\xi}) = \int_{\mathbb{R}^{m_0}} \mathbf{L}G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\,d\boldsymbol{\xi} \tag{5.38}$$

微分算子 $\mathbf{L}$ 将 $\boldsymbol{\xi}$ 视为常量，它作用于 $G(\mathbf{x},\boldsymbol{\xi})$时仅将其视为 $\mathbf{x}$ 的函数。将式(5.35)代入式(5.38)，有

$$\mathbf{L}F(\mathbf{x}) = \int_{\mathbb{R}^{m_0}} \delta(\mathbf{x} - \boldsymbol{\xi})\varphi(\boldsymbol{\xi})\,d\boldsymbol{\xi}$$

最后，利用 Dirac Delta 函数的筛选性质，可得

$$\int_{\mathbb{R}^{m_0}} \varphi(\boldsymbol{\xi})\delta(\mathbf{x} - \boldsymbol{\xi})\,d(\boldsymbol{\xi}) = \varphi(\mathbf{x})$$

这样我们就得到了如式(5.37)所描述的 $\mathbf{L}F(\mathbf{x})=\varphi(\mathbf{x})$。

正则化问题的解

回到当前的问题，下面我们来解 Euler-Lagrange 微分方程，即式(5.33)，令

$$\mathbf{L} = \tilde{\mathbf{D}}\mathbf{D} \tag{5.39}$$

和

$$\varphi(\boldsymbol{\xi}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\delta(\boldsymbol{\xi} - \mathbf{x}_i) \tag{5.40}$$

那么根据式(5.36)，有

$$\begin{aligned} F_\lambda(\mathbf{x}) &= \int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\left\{\frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\delta(\boldsymbol{\xi} - \mathbf{x}_i)\right\}d\boldsymbol{\xi} \\ &= \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\delta(\boldsymbol{\xi} - \mathbf{x}_i)\,d\boldsymbol{\xi} \end{aligned}$$

上式第二行交换了积分与求和的次序。最后，利用 Dirac Delta 函数的筛选性质，我们可以得到 Euler-Lagrange 微分方程(5.33)的解如下：

$$F_\lambda(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]G(\mathbf{x},\mathbf{x}_i) \tag{5.41}$$

272

式(5.41)说明正则化问题的最小化解 $F_\lambda(\mathbf{x})$是 N 个 Green 函数的线性叠加。$\mathbf{x}_i$ 代表扩展中心，权值$[d_i - F(\mathbf{x}_i)]/\lambda$ 代表展开系数。换句话说，正则化问题的解在光滑函数的空间的一个 N 维子空间上，以 $\mathbf{x}_i$，$i=1, 2, \cdots, N$ 为中心的一组 Green 函数$\{G(\mathbf{x},\mathbf{x}_i)\}$组成了该子空间的基(Poggio and Girosi, 1990a)。注意式(5.41)中，展开系数具有如下性质：(1)与系统的估计误差(定义为应有输出 d_i 和相应的网络实际计算输出 $F_i(\mathbf{x})$之差)成线性关系；(2)与正则化参数 λ 成反比。

确定展开系数

下面将要解决的问题是如何确定式(5.41)中的展开系数。令

$$w_i = \frac{1}{\lambda}[d_i - F(\mathbf{x}_i)],\ i = 1,2,\cdots,N \tag{5.42}$$

则正则化问题的最小化解(5.41)可以写成如下形式：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\mathbf{x},\mathbf{x}_i) \tag{5.43}$$

分别在 $\mathbf{x}_j$，$j=1, 2, \cdots, N$ 上计算式(5.43)的值，可得

$$F_\lambda(\mathbf{x}_j) = \sum_{i=1}^{N} w_i G(\mathbf{x}_j,\mathbf{x}_i),\ j = 1,2,\cdots,N \tag{5.44}$$

现在我们引入如下定义：

$$\mathbf{F}_\lambda = [F_\lambda(\mathbf{x}_1), F_\lambda(\mathbf{x}_2),\cdots,F_\lambda(\mathbf{x}_N)]^T \tag{5.45}$$

$$\mathbf{d} = [d_1, d_2, \cdots, d_N]^T \tag{5.46}$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{x}_1,\mathbf{x}_1) & G(\mathbf{x}_1,\mathbf{x}_2) & \cdots & G(\mathbf{x}_1,\mathbf{x}_N) \\ G(\mathbf{x}_2,\mathbf{x}_1) & G(\mathbf{x}_2,\mathbf{x}_2) & \cdots & G(\mathbf{x}_2,\mathbf{x}_N) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{x}_N,\mathbf{x}_1) & G(\mathbf{x}_N,\mathbf{x}_2) & \cdots & G(\mathbf{x}_N,\mathbf{x}_N) \end{bmatrix} \tag{5.47}$$

$$\mathbf{w} = [w_1, w_2, \cdots, w_N]^T \tag{5.48}$$

然后式(5.42)和(5.44)可分别写成矩阵形式

$$\mathbf{w} = \frac{1}{\lambda}(\mathbf{d} - \mathbf{F}_\lambda) \tag{5.49}$$

[273] 和
$$\mathbf{F}_\lambda = \mathbf{G}\mathbf{w} \tag{5.50}$$
消去式(5.49)和(5.50)中的 $\mathbf{F}_\lambda$，重新调整项我们可得
$$(\mathbf{G} + \lambda\mathbf{I})\mathbf{w} = \mathbf{d} \tag{5.51}$$
其中 $\mathbf{I}$ 是一个 $N \times N$ 阶的单位矩阵。矩阵 $\mathbf{G}$ 称为 Green 矩阵。

式(5.39)所定义的线性微分算子 $\mathbf{L}$ 是自伴的，它的伴随算子等于它自身。因此，与其相关的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 是对称函数，即对所有的 i，j 都有
$$G(\mathbf{x}_i,\mathbf{x}_j) = G(\mathbf{x}_j,\mathbf{x}_i) \tag{5.52}$$
式(5.52)表明 Green 函数 $G(\mathbf{x},\xi)$ 的两个自变量 $\mathbf{x}$ 和 ξ 的位置是可以互换的而不影响它的值。等价地，式(5.47)所定义的 Green 矩阵 $\mathbf{G}$ 是对称矩阵，即
$$\mathbf{G}^T = \mathbf{G} \tag{5.53}$$
现在我们回顾一下插值定理，它在 5.3 节中利用插值矩阵 $\mathbf{\Phi}$ 进行描述。我们首先注意到 Green 矩阵 $\mathbf{G}$ 在正则化理论中所起的作用与插值矩阵 $\mathbf{\Phi}$ 在 RBF 插值理论中所起的作用相同。它们都是 $N \times N$ 阶的对称阵。因此，我们可以说，对于某类 Green 函数，只要所提供的数据点 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$ 是互不相同的，则 Green 矩阵就是正定的。满足 Micchelli 定理的 Green 函数包括逆多二次函数和 Gauss 函数，但是没有多二次函数。实际上，我们总是将 λ 选得足够大，使得 $\mathbf{G}+\lambda\mathbf{I}$ 是正定的，从而是可逆的。这样式(5.51)所表示的线性方程组就具有惟一解(Poggio and Girosi, 1990a)：
$$\mathbf{w} = (\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{d} \tag{5.54}$$
因此，只要选定了微分算子 $\mathbf{D}$，从而确定了相应的 Green 函数 $G(\mathbf{x}_j,\mathbf{x}_i)$，$i=1, 2, \cdots, N$，我们就可以通过计算式(5.54)得到与某一特定期望输出向量 $\mathbf{d}$ 以及合适的正则化参数值 λ 相对应的权值向量 $\mathbf{w}$。

总之，我们可以说正则化问题的解可以由展开式[9]
$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\mathbf{x},\mathbf{x}_i) \tag{5.55}$$
给出，其中 $G(\mathbf{x},\mathbf{x}_i)$ 是自伴微分算子 $\mathbf{L}=\tilde{\mathbf{D}}\mathbf{D}$ 的 Green 函数，w_i 是权值向量 $\mathbf{w}$ 的第 i 个元素，这两个量分别由式(5.53)和式(5.54)定义。由式(5.55)可知(Poggio and Girosi, 1990a)：

- 正则化方法等价于在一组 Green 函数的基础上解的展开，它们的特性只决定于所采用的稳定因子 $\mathbf{D}$ 的形式和相关的边界条件。
- 在展开式中所用到的 Green 函数的个数与训练过程中所用的样本数据点的个数相同。

但是应该注意的是，式(5.55)所给出的正则化问题的解是不完整的，因为它代表一个对位于算子 $\mathbf{D}$ 的零空间上项 $g(\mathbf{x})$ 的解的模(Poggio and Girosi, 1990a)。我们这么说是因为所有位于 $\mathbf{D}$ 的零空间上的函数对于式(5.23)的目标泛函 $\mathscr{E}(F)$ 中的 $\|\mathbf{D}F\|^2$ 项都是“不可见”的。[274] 我们所说 $\mathbf{D}$ 的零空间是指所有满足 $\mathbf{D}g$ 等于零的函数 $g(\mathbf{x})$ 的集合。附加项 $g(\mathbf{x})$ 的确切形式是依赖问题的，也就是它取决于问题的稳定因子的选取以及边界条件。例如，当稳定因子 $\mathbf{D}$ 对应于一个钟形 Green 函数，如 Gauss 函数或者逆多二次函数，此时就不需要 $g(\mathbf{x})$。由于这个原因，并且它的存在并不会对最后主要结果产生影响，所以我们在结果中忽略这个问题。

对于某一特定的中心 $\mathbf{x}_i$，Green 函数的特性只取决于所选的稳定因子，即只取决于关于输入－输出映射的先验假设。如果所选的稳定因子 $\mathbf{D}$ 具有平移不变性，则以 $\mathbf{x}_i$ 为中心的

Green 函数 $G(\mathbf{x}, \mathbf{x}_i)$只取决于自变量 $\mathbf{x}$ 和 $\mathbf{x}_i$ 之差；即

$$G(\mathbf{x},\mathbf{x}_i) = G(\mathbf{x}-\mathbf{x}_i) \tag{5.56}$$

如果稳定因子 $\mathbf{D}$ 是平移不变和旋转不变的，则 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$只取决于向量 $\mathbf{x}-\mathbf{x}_i$ 的 Euclid 范数，表示为

$$G(\mathbf{x},\mathbf{x}_i) = G(\|\mathbf{x}-\mathbf{x}_i\|) \tag{5.57}$$

在这些条件下，Green 函数一定是径向基函数。此时，式(5.55)的正则化问题的解可表示为如下形式(Poggio and Girosi, 1990a)：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\|\mathbf{x}-\mathbf{x}_i\|) \tag{5.58}$$

式(5.58)所描述的解构造一个依赖于已知数据点的 Euclid 距离度量的线性函数空间。

式(5.58)所描述的解叫做严格插值解，因为所有 N 个已知训练数据点都被用于生成插值函数 $F(\mathbf{x})$。但是，值得注意的是式(5.58)与式(5.11)所表示的解有根本不同：式(5.58)的解被式(5.54)给出的权重向量 $\mathbf{w}$ 的定义所正则化。只有当我们将正则化参数 λ 设为零时，这两个解才是一样的。

多元 Gauss 函数

Green 函数 $G(\mathbf{x},\mathbf{x}_i)$的相应的线性微分算子 $\mathbf{D}$ 是平移不变和旋转不变的并且它满足式(5.57)的条件，此时 Green 函数具有重要实际意义。这类 Green 函数的一个例子是多元 Gauss 函数，定义为

$$G(\mathbf{x},\mathbf{x}_i) = \exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right) \tag{5.59}$$

275

其中 $\mathbf{x}_i$ 表示函数的中心，而 σ_i 则表示它的宽度。与式(5.59)所示 Green 函数相对应的自伴随算子 $\mathbf{L}=\tilde{\mathbf{D}}\mathbf{D}$ 由下式给出(Poggio and Girosi, 1990a)：

$$\mathbf{L} = \sum_{n=0}^{\infty}(-1)^n \alpha_n \nabla^{2n} \tag{5.60}$$

其中

$$\alpha_n = \frac{\sigma_i^{2n}}{n!\,2^n} \tag{5.61}$$

而∇^{2n}是 m_0 维多重拉普拉斯算子

$$\nabla^2 = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \cdots + \frac{\partial^2}{\partial x_{m_0}^2} \tag{5.62}$$

因为式(5.60)中 $\mathbf{L}$ 的项数允许到无穷，所以从标准意义上说 $\mathbf{L}$ 并不是一个微分算子。因此，我们将式(5.60)中的 $\mathbf{L}$ 称为伪微分算子。

由于定义 $\mathbf{L}=\tilde{\mathbf{D}}\mathbf{D}$，由式(5.60)我们可以推导出算子 $\mathbf{D}$ 和 $\tilde{\mathbf{D}}$ 如下(参见注释[10])：

$$\mathbf{D} = \sum_n \alpha_n^{1/2}\left(\frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \cdots + \frac{\partial}{\partial x_{m_0}}\right)^n = \sum_{a+b+\cdots+k=n} \alpha_n^{1/2}\frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k} \tag{5.63}$$

和

$$\tilde{\mathbf{D}} = \sum_n (-1)^n \alpha_n^{1/2}\left(\frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \cdots + \frac{\partial}{\partial x_{m_0}}\right)^n = \sum_{a+b+\cdots+k=n} (-1)\alpha_n^{1/2}\frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k} \tag{5.64}$$

因此通过使用包括所有可能偏导数在内的稳定因子，可以得到式(5.58)形式的正则解。

将式(5.59)至(5.61)代入式(5.35)且令 ξ 为 $\mathbf{x}_i$，则我们有

$$\sum_n (-1)^n \frac{\sigma_i^{2n}}{n!2^n} \nabla^{2n} \exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right) = \delta(\mathbf{x}-\mathbf{x}_i) \tag{5.65}$$

利用(5.59)定义的 Green 函数的特殊形式，我们就可以将式(5.55)给出的正则化解写成多元 Gauss 函数的线性叠加形式如下：

276

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i \exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right) \tag{5.66}$$

其中线性权值 w_i 由式(5.42)定义。

在式(5.66)中，定义逼近函数 $F(\mathbf{x})$的各 Gauss 项的方差是不同的。为简化起见，通常认为在 $F(\mathbf{x})$中对所有的 i 都有 $\sigma_i=\sigma$ 。尽管这样设计的 RBF 网络是受到一定限制的一种，但其仍不失为一个通用逼近器(Park and Sandberg, 1991)。

5.6 正则化网络

式(5.55)给出的正则化逼近函数 $F_\lambda(\mathbf{x})$关于中心在 $\mathbf{x}_i$ 的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$的展开预示着图 5-4 所示网络结构为其提供一个实现方法。基于明显的原因，这种网络结构被称为正则化网络(Poggio and Girosi, 1990a)。如 5.1 节所述的网络一样，该网络包括三层。第一层是由输入节点组成的，输入节点数目等于输入向量 $\mathbf{x}$ 的维数 m_0(即问题的独立变量数)。第二层是隐藏层，它是由直接与所有输入节点相连的非线性单元组成的。一个隐藏单元对应一个数据点 $\mathbf{x}_i$，$i=1, 2, \cdots, N$，其中 N 表示训练样本的长度。每个隐藏单元的激活函数由 Green 函数定义。由此第 i 个隐藏单元的输出是 $G(\mathbf{x},\mathbf{x}_i)$。输出层仅包括一个线性单元，它与所有隐藏单元相连。这里所谓的"线性"指的是网络的输出是隐藏单元输出的线性加权和。输出层的权值就是未知的展开系数，如式(5.54)所示，它是由 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$和正则化参数 λ 决定。图 5-4 描绘一个单输出的正则化网络的结构图。显然，我们可以将其推广为包括任意期望输出数目的正则化网络。

277

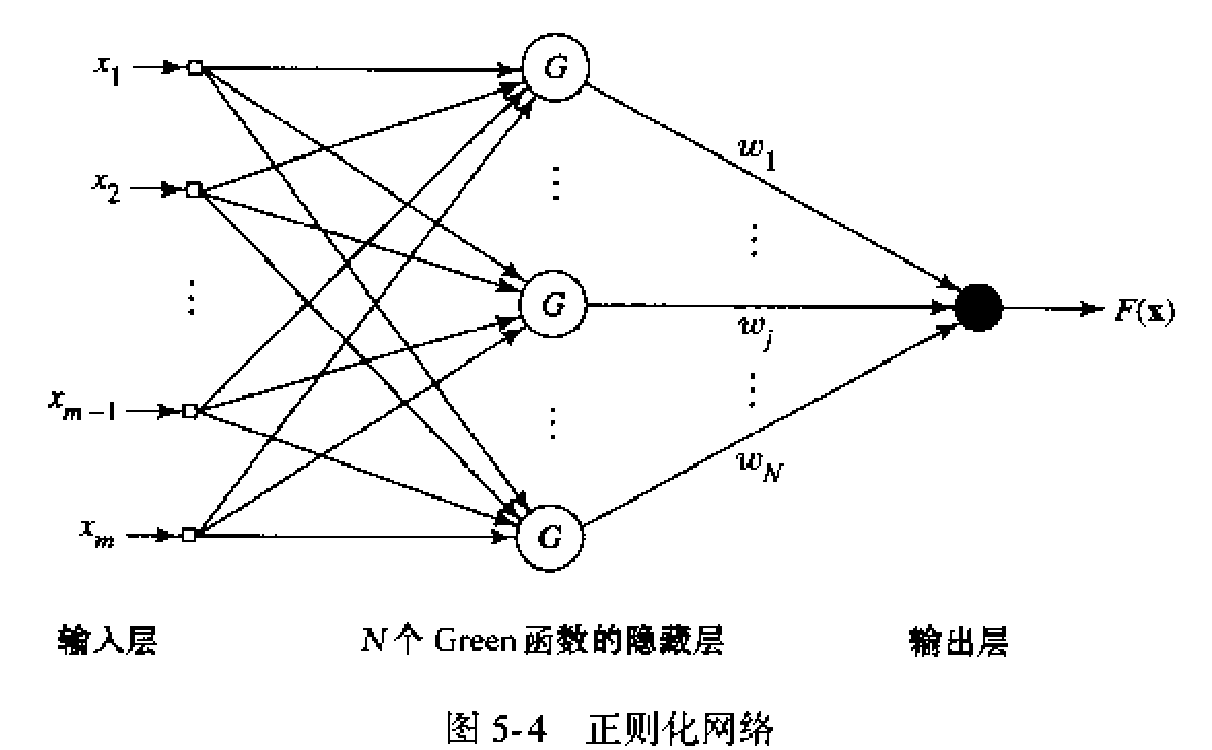


图 5-4 正则化网络

图 5-4 所示的正则化网络假设 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$对所有的 i 都是正定的。假设上述条件成立，例如，Green 函数具有式(5.59)所示 Gauss 形式，则由该网络所得到的解在泛函$\mathscr{E}(F)$最小

化的意义下将是一个"最佳"的内插解。而且，由逼近理论的观点，正则化网络具有如下三个期望的性质(Poggio and Girosi,1990a)：

1. 正则化网络是一个通用逼近器，只要有足够多的隐藏单元，它可以以任意精度逼近定义在 $\mathbb{R}^{m_0}$ 的紧子集上的任何多元连续函数。

2. 由于正则化理论导出的逼近格式的未知系数是线性的，这样该网络具有最佳逼近性能。这说明给定一个未知的非线性函数 f，总可选择一组系数使得它对 f 的逼近优于所有其他可能选择。

3. 由正则化网络求得的解是最佳的。这里的最佳是指正则化网络使测量训练样本表示的解与真实值有多大偏差的泛函最小化。

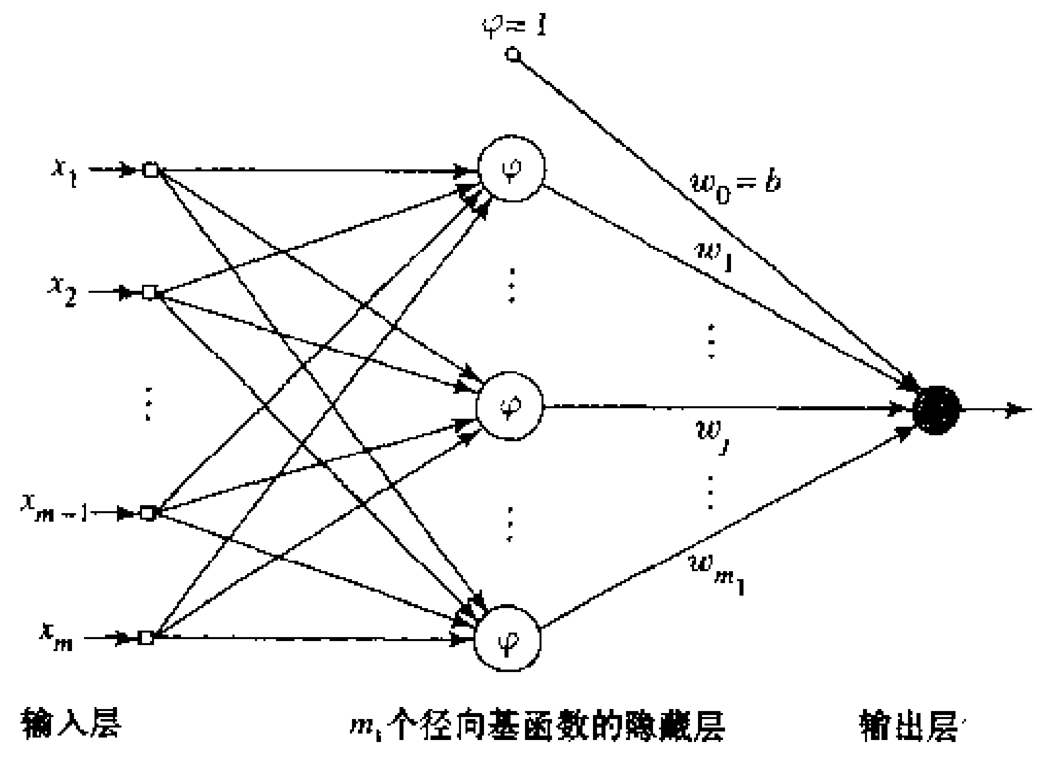


图 5-5 径向基函数网络

5.7 广义径向基函数网络

由于输入向量 $\mathbf{x}_i$ 与 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$，$i=1, 2, \cdots, N$ 之间的一一对应的关系，有时候如果 N 太大了，实现它的计算量将大得惊人。特别是在计算网络的线性权值(即式(5.55)中的展开系数)时，要求计算一个 $N\times N$ 阶矩阵的逆，其计算量按 N 的多项式增长(大约为 N^3)。另外矩阵越大，其病态的可能性越高；一个矩阵的条件数被定义为该矩阵的最大特征值与其最小特征值的比值。为了克服这些计算上的困难，我们通常要降低神经网络的复杂度，这要求一个正则化解的近似。 278

解决办法是在一个较低维数的空间中求一个次优解，以此来逼近式(5.55)所给出的正则化解。这可以通过变分问题中通称 Galerkin 方法的标准技术实现。根据这个技术，近似解 $F^*(\mathbf{x})$ 将在一个有限基上进行扩展，表示为(Poggio and Girosi,1990a)

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i\varphi_i(\mathbf{x}) \tag{5.67}$$

其中 $\{\varphi_i(\mathbf{x})\mid i=1,2,\cdots,m_1\}$ 是一组新的基函数，不失一般性我们假设它们线性独立。典型情况下这组新的基函数的个数小于输入数据点的个数(即 $m_1\leqslant N$)，并且 w_i 组成一组新的权值集合。根据径向基函数，我们设

$$\varphi_i(\mathbf{x}) = G(\|\mathbf{x}-\mathbf{t}_i\|), \qquad i=1,2,\cdots,m_1 \tag{5.68}$$

其中中心集 $\{\mathbf{t}_i\mid i=1,2,\cdots,m_1\}$ 待定。基函数的这个特定选择是惟一的选择，使得能保证当 $m_1=N$，且 $\mathbf{t}_i=\mathbf{x}_i$，$i=1, 2, \cdots, N$ 时，其解与式(5.58)的正确解一致。因此将式(5.68)代入式(5.67)中，我们可以重新定义 $F^*(\mathbf{x})$ 为

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_iG(\mathbf{x},\mathbf{t}_i) = \sum_{i=1}^{m_1} w_iG(\|\mathbf{x}-\mathbf{t}_i\|) \tag{5.69}$$

给定逼近函数 $F^*(\mathbf{x})$ 的(5.69)的展开形式，我们将要解决的问题是确定一组新的权值 $\{w_i\mid i=1,2,\cdots,m_1\}$，使新的代价泛函 $\mathscr{E}(F^*)$ 最小化，新代价泛函由

$$\mathscr{E}(F^*) = \sum_{i=1}^{N}\left(d_i - \sum_{j=1}^{m_1} w_j G(\|\mathbf{x}_i - \mathbf{t}_j\|)\right)^2 + \lambda\|\mathbf{D}F^*\|^2 \tag{5.70}$$

定义。式(5.70)右边第一项可以写成欧几里德范数平方$\|\mathbf{d}-\mathbf{G}\mathbf{w}\|^2$，其中

$$\mathbf{d} = [d_1, d_2, \cdots, d_N]^T \tag{5.71}$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{x}_1,\mathbf{t}_1) & G(\mathbf{x}_1,\mathbf{t}_2) & \cdots & G(\mathbf{x}_1,\mathbf{t}_{m_1}) \\ G(\mathbf{x}_2,\mathbf{t}_1) & G(\mathbf{x}_2,\mathbf{t}_2) & \cdots & G(\mathbf{x}_2,\mathbf{t}_{m_1}) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{x}_N,\mathbf{t}_1) & G(\mathbf{x}_N,\mathbf{t}_2) & \cdots & G(\mathbf{x}_N,\mathbf{t}_{m_1}) \end{bmatrix} \tag{5.72}$$

279

$$\mathbf{w} = [w_1, w_2, \cdots, w_{m_1}]^T \tag{5.73}$$

期望响应向量$\mathbf{d}$与前面一样是N维的。但是，Green函数的矩阵$\mathbf{G}$和权值向量$\mathbf{w}$的维数却有不同维数；矩阵$\mathbf{G}$现在是$N\times m_1$阶的，所以不再是对称的，而向量$\mathbf{w}$是$m_1\times 1$的。由式(5.69)我们注意到，近似函数F^*是由稳定因子$\mathbf{D}$决定的Green函数的线性组合。因此，我们可以将式(5.70)右边第二项写成

$$\begin{aligned}\|\mathbf{D}F^*\|^2 &= (\mathbf{D}F^*, \mathbf{D}F^*)_{\mathscr{H}} = \left[\sum_{i=1}^{m_1} w_i G(\mathbf{x},\mathbf{t}_i), \tilde{\mathbf{D}}\mathbf{D}\sum_{i=1}^{m_1} w_i G(\mathbf{x};\mathbf{t}_i)\right]_{\mathscr{H}} \\ &= \left[\sum_{i=1}^{m_1} w_i G(\mathbf{x},\mathbf{t}_i), \sum_{i=1}^{m_1} w_i \delta_{\mathbf{t}_i}\right]_{\mathscr{H}} = \sum_{j=1}^{m_1}\sum_{i=1}^{m_1} w_j w_i G(\mathbf{t}_j,\mathbf{t}_i) = \mathbf{w}^T\mathbf{G}_0\mathbf{w}\end{aligned} \tag{5.74}$$

其中第二个和第三个相等项分别利用伴随算子的定义和式(5.35)。矩阵$\mathbf{G}_0$是一个$m_1\times m_1$阶的对称阵，定义为

$$\mathbf{G}_0 = \begin{bmatrix} G(\mathbf{t}_1,\mathbf{t}_1) & G(\mathbf{t}_1,\mathbf{t}_2) & \cdots & G(\mathbf{t}_1,\mathbf{t}_{m_1}) \\ G(\mathbf{t}_2,\mathbf{t}_1) & G(\mathbf{t}_2,\mathbf{t}_2) & \cdots & G(\mathbf{t}_2,\mathbf{t}_{m_1}) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{t}_{m_1},\mathbf{t}_1) & G(\mathbf{t}_{m_1},\mathbf{t}_2) & \cdots & G(\mathbf{t}_{m_1},\mathbf{t}_{m_1}) \end{bmatrix} \tag{5.75}$$

以权值向量$\mathbf{w}$为变量求式(5.70)的最小值，可以得到以下结果(参看习题5.5)：

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_0)_{\mathbf{w}} = \mathbf{G}^T\mathbf{d} \tag{5.76}$$

当正则化参数λ趋近零时，权值向量$\mathbf{w}$趋于一个超定的最小平方数据－拟合问题(因为$m_1 < N$)的伪逆(最小范数)解，表示为(Broomhead and Lowe, 1988)

$$\mathbf{w} = \mathbf{G}^+\mathbf{d}, \lambda = 0 \tag{5.77}$$

其中$\mathbf{G}^+$是矩阵$\mathbf{G}$的伪逆；即

$$\mathbf{G}^+ = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \tag{5.78}$$

加权范数

式(5.69)中的范数通常指的是欧几里德范数。然而，当输入向量$\mathbf{x}$的分量属于不同的类时，将其视为一般的*加权范数*会更合理，加权范数的平方形式由

280

$$\|\mathbf{x}\|_C^2 = (\mathbf{C}\mathbf{x})^T(\mathbf{C}\mathbf{x}) = \mathbf{x}^T\mathbf{C}^T\mathbf{C}\mathbf{x} \tag{5.79}$$

定义(Poggio and Girosi, 1990a)，其中$\mathbf{C}$是一个$m_0\times m_0$加权矩阵，m_0是输入向量$\mathbf{x}$的维数。

利用加权范数的定义，我们可以将式(5.69)中正则化问题的近似解写成如下更一般的形式(Lowe,1989;Poggio and Girosi,1990a)：

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\|\mathbf{x}-\mathbf{t}_i\|_C) \tag{5.80}$$

引入加权范数可以用两种方式解释。我们可以简单将其视为对原始输入空间做一个仿射变换。原则上这种变换并不会降低原来不加权的结果，因为原来不加权的范数实际上对应于一个单位矩阵的加权范数。另一方面，加权范数可以看作直接从式(5.63)定义的 m_0 维 Laplace 伪微分算子 $\mathbf{D}$ 的少许推广；参见习题 5.6。使用加权范数的合理性在 Gauss 径向基函数背景下可以解释如下。一个以 $\mathbf{t}_i$ 为中心和具有范数加权矩阵 $\mathbf{C}$ 的 Gauss 径向基函数 $G(\|\mathbf{x}-\mathbf{t}_i\|_C)$可写成

$$\begin{aligned} G(\|\mathbf{x}-\mathbf{t}_i\|_C) &= \exp[-(\mathbf{x}-\mathbf{t}_i)^T\mathbf{C}^T\mathbf{C}(\mathbf{x}-\mathbf{t}_i)] \\ &= \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{t}_i)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{t}_i)\right] \end{aligned} \tag{5.81}$$

其中逆矩阵 $\mathbf{\Sigma}^{-1}$定义为

$$\frac{1}{2}\mathbf{\Sigma}^{-1} = \mathbf{C}^T\mathbf{C} \tag{5.82}$$

式(5.81)表示一个具有均值向量 $\mathbf{t}_i$ 和协方差矩阵 $\mathbf{\Sigma}$ 的多元 Gauss 分布。基于此，它是式(5.59)描述分布的推广。

式(5.70)中逼近问题的解为具有如图 5-5 结构的广义径向基函数网络提供了一个框架。在这种网络中，输出单元上有一个偏置(即独立于数据的变量)。要做到这一点可以简单将输出层的一个线性权值置为偏置值，同时将与该权值相对应的径向基函数视为一个等于 +1 的常量。

从结构上看，图 5-5 所示的广义 RBF 网络与图 5-4 所示的正则化 RBF 网络相似。但它们在以下两个重要的方面不同：

1. 图 5-5 所示的广义 RBF 网络隐藏层的节点数为 m_1，通常 m_1 总是小于用于训练的样本数 N。另一方面，图 5-4 所示的正则化 RBF 网络的隐藏单元数恰为 N。

2. 在图 5-5 的广义 RBF 网络中，与输出层相连的线性权值向量，以及与隐藏层相连的径向基函数的中心和范数加权矩阵，均为待学习的未知参数。而图 5-4 的正则化 RBF 网络隐藏层的激活函数是已知的，它定义为一组以训练样本点为中心的 Green 函数；输出层的权值向量是网络的惟一的未知参数。 [281]

接受域

协方差矩阵 $\mathbf{\Sigma}$ 决定式(5.81)给出的 Gauss 径向基函数 $G(\|\mathbf{x}-\mathbf{t}_i\|_C)$的接受域。给定一个中心 $\mathbf{t}_i$，$G(\|\mathbf{x}-\mathbf{t}_i\|_C)$的接受域形式地定义为函数

$$\Psi(\mathbf{x}) = G(\|\mathbf{x}-\mathbf{t}_i\|_C) - a \tag{5.83}$$

的支集，其中 a 是一个正常数(Xu et al.,1994)。换句话说，$G(\|\mathbf{x}-\mathbf{t}_i\|_C)$的接受域是输入向量 $\mathbf{x}$ 的定义域的一个特殊子集，这个子集中的所有 $\mathbf{x}$ 都能使 $G(\|\mathbf{x}-\mathbf{t}_i\|_C)$取值大于给定水平 a。

根据加权范数矩阵 $\mathbf{C}$ 的不同定义方式，我们可以分三种情况讨论协方差矩阵 $\mathbf{\Sigma}$ 及其对

接受域的形状、大小和方向的影响：

1. $\mathbf{\Sigma} = \sigma^2\mathbf{I}$，其中 $\mathbf{I}$ 是单位矩阵，σ^2 是公共方差。此时，$G(\|\mathbf{x}-\mathbf{t}_i\|_C)$ 的接受域是以 $\mathbf{t}_i$ 为中心和半径由 σ 决定的超球面。

2. $\mathbf{\Sigma} = \text{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_{m_0}^2)$，其中 σ_j^2 是输入向量 $\mathbf{x}$ 的第 j 个分量的方差，$j=1, 2, \cdots, m_0$。在这种情形，$G(\|\mathbf{x}-\mathbf{t}_i\|_C)$ 的接受域是一个超椭圆面，它的轴与输入空间的轴一致，沿第 j 个轴的伸延由 σ_j 决定。

3. $\mathbf{\Sigma}$ 是一个非对角矩阵。根据定义，$\mathbf{\Sigma}$ 是一个正定矩阵。所以我们可以用矩阵代数中的相似变换来分解 $\mathbf{\Sigma}$ 如下：

282

$$\mathbf{\Sigma} = \mathbf{Q}^T\mathbf{\Lambda}\mathbf{Q} \tag{5.84}$$

其中 $\mathbf{\Lambda}$ 是一个对角矩阵，而 $\mathbf{Q}$ 是一个正交旋转矩阵。矩阵 $\mathbf{\Lambda}$ 决定接受域的形状和大小，而矩阵 $\mathbf{Q}$ 决定接受域的方向。

5.8　XOR 问题(再讨论)

再考虑第4章中我们用单隐藏层的多层感知器模型解决过的 XOR(异或)问题。这里我们将给出用 RBF 网络求解这个问题的解。

被研究的 RBF 网络由一对 Gauss 函数组成，它们定义如下：

$$G(\|\mathbf{x}-\mathbf{t}_i\|) = \exp(-\|\mathbf{x}-\mathbf{t}_i\|^2), i = 1,2 \tag{5.85}$$

其中中心 $\mathbf{t}_1$ 和 $\mathbf{t}_2$ 为

$$\mathbf{t}_1 = [1,1]^T, \mathbf{t}_2 = [0,0]^T$$

对输出单元的特性，我们作如下假设：

1. 由于问题是对称的，输出单元使用权值共享，这是先验知识嵌入网络设计的一种形式。因此，虽然有两个隐藏单元，我们只有一个权值 w 有待确定。

2. 输出单元包括一个偏置 b(即独立于数据的变量)。此偏置的作用是保证 XOR 函数具有非零均值的输出值。

用于解决 XOR 问题的 RBF 网络结构如图 5-6 所示。该网络的输入输出关系可定义为

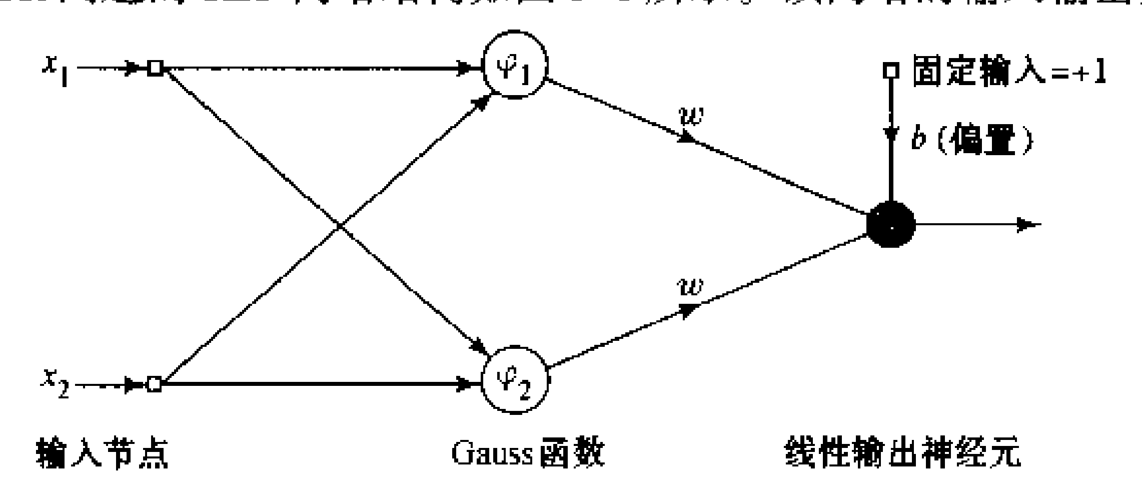


图 5-6　求解 XOR 问题的 RBF 网络

$$y(\mathbf{x}) = \sum_{i=1}^{2} wG(\|\mathbf{x}-\mathbf{t}_i\|) + b \tag{5.86}$$

为了拟合表 5-2 所示的训练数据，我们要求

$$y(\mathbf{x}_j) = d_j, \quad j = 1,2,3,4 \tag{5.87}$$

其中 $\mathbf{x}_j$ 是输入向量，d_j 是与其相应的期望输出值。令

$$g_{ji} = G(\| \mathbf{x}_j - \mathbf{t}_i \|), \qquad j = 1,2,3,4; i = 1,2 \tag{5.88}$$

利用表 5-2 的值代入式(5.88)，我们可以得到如下以矩阵形式表示的方程组：

283

$$\mathbf{G}\mathbf{w} = \mathbf{d} \tag{5.89}$$

表 5-2 XOR 问题的输入 - 输出变换计算

数据点 j	输入模式 $\mathbf{x}_j$	期望输出 d_j
1	(1,1)	0
2	(0,1)	1
3	(0,0)	0
4	(1,0)	1

其中

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1353 & 1 \\ 0.3678 & 0.3678 & 1 \\ 0.1353 & 1 & 1 \\ 0.3678 & 0.3678 & 1 \end{bmatrix} \tag{5.90}$$

$$\mathbf{d} = [0 \quad 1 \quad 0 \quad 1]^T \tag{5.91}$$

$$\mathbf{w} = [w \quad w \quad b]^T \tag{5.92}$$

这里描述的问题是*超定的*，这是就数据点的个数比自由参数数目多的意义而言的。这就解释矩阵 $\mathbf{G}$ 为什么不是方阵的原因。因此，矩阵 $\mathbf{G}$ 不存在惟一的逆。为了克服这个困难，我们用式(5.78)的最小范数解来解决这个问题，由此可得

$$\mathbf{w} = \mathbf{G}^{+}\mathbf{d} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{d} \tag{5.93}$$

注意 $\mathbf{G}^T\mathbf{G}$ 是一个方阵，其逆存在。将式(5.90)代入式(5.93)，我们有

$$\mathbf{G}^{+} = \begin{bmatrix} 1.8292 & -1.2509 & 0.6727 & -1.2509 \\ 0.6727 & -1.2509 & 1.8292 & -1.2509 \\ -0.9202 & 1.4202 & -0.9202 & 1.4202 \end{bmatrix} \tag{5.94}$$

最后，将式(5.91)和式(5.94)都代入式(5.93)中，可得

$$\mathbf{w} = \begin{bmatrix} -2.5018 \\ -2.5018 \\ +2.8404 \end{bmatrix}$$

这样，我们就用 RBF 网络完整解决了 XOR 问题。

5.9 正则化参数估计

284

正则化参数 λ 在 5.5 节至 5.7 节提出的径向基函数网络正则化理论中起着中心的作用。为了更好的利用这个理论，我们需要一个估计 λ 的相当于原理性的方法。

为了形成我们的思想，先考虑一个*非线性回归问题*，它由一个模型描述，其中与第 i 时间步的输入向量 $\mathbf{x}_i$ 相对应的可观测输出 y_i 定义为

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1,2,\cdots,N \tag{5.95}$$

此处 $f(\mathbf{x}_i)$是一条"光滑曲线"，ε_i 是一个均值为零和方差为 σ^2 的白噪声过程的采样。即

$$E[\varepsilon_i] = 0 \qquad 对所有\ i \tag{5.96}$$

和
$$E[\varepsilon_i\varepsilon_k]=\begin{cases}\sigma^2 & \text{对于 } k=i\\ 0 & \text{其他}\end{cases} \tag{5.97}$$
问题是在给定一组训练样本$\{(\mathbf{x}_i, y_i)\}_{i=1}^N$的条件下，重建该模型的固有函数$f(\mathbf{x}_i)$。

令$F_\lambda(\mathbf{x})$为$f(\mathbf{x})$相对于某个正则化参数λ的正则化估计。即$F_\lambda(\mathbf{x})$为使表示非线性回归问题的 Tikhonov 泛函
$$\mathscr{E}(F)=\frac{1}{2}\sum_{i=1}^N[y_i-F(\mathbf{x}_i)]^2+\frac{\lambda}{2}\|\mathbf{D}F(\mathbf{x})\|^2 \tag{5.98}$$
达到最小的最小化函数。选择一个合适的λ值并不是一个简单事，它需要在下面两种矛盾的情况之间加以权衡：

- 由$\|\mathbf{D}F(\mathbf{x})\|^2$项来度量解的粗糙度
- 由$\sum_{i=1}^N[y_i-F(\mathbf{x}_i)]^2$项来度量数据的失真度

这一节的主题是讨论如何选择好的正则化参数λ。

均方误差

令$R(\lambda)$表示模型的回归函数$f(\mathbf{x})$和表示在正则化参数λ某一值下的解的逼近函数$F_\lambda(\mathbf{x})$之间在整个给定集合上的均方误差。即
$$R(\lambda)=\frac{1}{N}\sum_{i=1}^N[f(\mathbf{x}_i)-F_\lambda(\mathbf{x}_i)]^2 \tag{5.99}$$
所谓最佳λ指的是使$R(\lambda)$取最小的λ值。

将$F_\lambda(\mathbf{x}_k)$表示为给定的一组可观察值的线性组合：
$$F_\lambda(\mathbf{x}_k)=\sum_{i=1}^N a_{ki}(\lambda)y_i \tag{5.100}$$
用等价的矩阵形式写成

285

$$\mathbf{F}_\lambda=\mathbf{A}(\lambda)\mathbf{y} \tag{5.101}$$
其中
$$\mathbf{F}_\lambda=[F_\lambda(\mathbf{x}_1),F_\lambda(\mathbf{x}_2),\cdots,F_\lambda(\mathbf{x}_N)]^T$$
$$\mathbf{y}=[y_1,y_2,\cdots,y_N]^T$$
且
$$\mathbf{A}(\lambda)=\begin{bmatrix}a_{11} & a_{12} & \cdots & a_{1N}\\ a_{21} & a_{22} & \cdots & a_{2N}\\ \vdots & \vdots & & \vdots\\ a_{N1} & a_{N2} & \cdots & a_{NN}\end{bmatrix} \tag{5.102}$$
其中$N\times N$矩阵$\mathbf{A}(\lambda)$称为*影响矩阵*。

用上述的矩阵符号，我们可将式(5.99)重新写成
$$R(\lambda)=\frac{1}{N}\|\mathbf{f}-\mathbf{F}_\lambda\|^2=\frac{1}{N}\|\mathbf{f}-\mathbf{A}(\lambda)\mathbf{y}\|^2 \tag{5.103}$$
其中$N\times 1$的向量$\mathbf{f}$为
$$\mathbf{f}=[f(\mathbf{x}_1),f(\mathbf{x}_2),\cdots,f(\mathbf{x}_N)]^T$$
我们可以进一步将式(5.95)也写成矩阵形式
$$\mathbf{y}=\mathbf{f}+\boldsymbol{\varepsilon} \tag{5.104}$$

其中
$$\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_N]^T$$
因此，将式(5.104)代入式(5.103)中并展开，可得

$$\begin{aligned} R(\lambda) &= \frac{1}{N}\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} - \mathbf{A}(\lambda)\boldsymbol{\varepsilon}\|^2 \\ &= \frac{1}{N}\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f}\|^2 - \frac{2}{N}\boldsymbol{\varepsilon}^T\mathbf{A}(\lambda)(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} + \frac{1}{N}\|\mathbf{A}(\lambda)\boldsymbol{\varepsilon}\|^2 \end{aligned} \tag{5.105}$$

其中 $\mathbf{I}$ 是一个 $N \times N$ 的单位矩阵。为求 $R(\lambda)$的期望值，注意下述几点： [286]

- 式(5.105)的右边第一项是一个常数，因此它不受期望算子的影响。
- 由式(5.96)可知，第二项的期望为零。
- 纯量 $\|\mathbf{A}(\lambda)\boldsymbol{\varepsilon}\|^2$ 的期望为

$$\begin{aligned} E[\|\mathbf{A}(\lambda)\boldsymbol{\varepsilon}\|^2] &= E[\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}] \\ &= \mathrm{tr}\{E[\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}]\} = E\{\mathrm{tr}[\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}]\} \end{aligned} \tag{5.106}$$

其中我们首先用到了纯量的迹等于纯量本身的性质，然后交换了期望运算和求迹运算的次序。

接下来我们利用矩阵代数中的如下规则：给定两个具有相容维数的矩阵 $\mathbf{B}$ 和 $\mathbf{C}$，$\mathbf{BC}$ 的迹等于 $\mathbf{CB}$ 的迹。令 $\mathbf{B} = \boldsymbol{\varepsilon}^T$，$\mathbf{C} = \mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}$，则式(5.106)可以写成等价形式

$$E[\|\mathbf{A}(\lambda)\mathbf{f}\|^2] = E\{\mathrm{tr}[\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T]\} = \sigma^2\mathrm{tr}[\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)] \tag{5.107}$$

上式中的最后一行根据式(5.97)可得。最后注意到 $\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)$的迹等于 $\mathbf{A}^2(\lambda)$的迹，则

$$E[\|\mathbf{A}(\lambda)\mathbf{f}\|^2] = \sigma^2\mathrm{tr}[\mathbf{A}^2(\lambda)] \tag{5.108}$$

将这三项结果结合起来，$R(\lambda)$期望值可表示为

$$E[R(\lambda)] = \frac{1}{N}\|\mathbf{I} - \mathbf{A}(\lambda)\mathbf{f}\|^2 + \frac{\sigma^2}{N}\mathrm{tr}[\mathbf{A}^2(\lambda)] \tag{5.109}$$

但是，一个给定数据集的均方误差 $R(\lambda)$在实际中并不好用，因为式(5.109)中需要回归函数 $f(\mathbf{x})$的知识，它是有待重建的函数。我们引入如下定义作为 $R(\lambda)$的估计(Craven and Wahba, 1979)：

$$\hat{R}(\lambda) = \frac{1}{N}\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2 + \frac{\sigma^2}{N}\mathrm{tr}[\mathbf{A}^2(\lambda)] - \frac{\sigma^2}{N}\mathrm{tr}[(\mathbf{I} - \mathbf{A}(\lambda))^2] \tag{5.110}$$

它是无偏估计，因此(按照导出式(5.109)所述的相似过程)我们可证明

$$E[\hat{R}(\lambda)] = E[R(\lambda)] \tag{5.111}$$

所以，使估计 $\hat{R}(\lambda)$最小的 λ 值可以作为正则化参数 λ 的一个好的选择。

广义交叉确认

使用估计 $\hat{R}(\lambda)$的一个缺陷是它要求知道噪声的方差 σ^2。在实际情况中，σ^2 通常是未知的。为了处理这种情况，下面我们将介绍广义交叉确认，它最早是由 Craven and Wahba (1979)提出的。 [287]

我们从修改通常的交叉确认的留一形式(在第 4 章描述)开始来处理这个问题。具体地，令 $F_\lambda^{[k]}(\mathbf{x})$为使泛函

$$\mathscr{E}(F) = \frac{1}{2}\sum_{\substack{i=1 \\ i \neq k}}^{N}[y_i - F_\lambda(\mathbf{x}_i)]^2 + \frac{\lambda}{2}\|\mathbf{D}F(\mathbf{x})\|^2 \tag{5.112}$$

最小化的函数，其中标准误差项中省略了第 k 项 $[y_k - F_\lambda(\mathbf{x}_k)]$。通过留出这一项，我们将用 $F_\lambda^{[k]}(\mathbf{x})$ 预报缺损数据点 y_k 的能力来衡量参数 λ 的好坏。因此，我们可以引入性能度量

$$V_0(\lambda) = \frac{1}{N}\sum_{k=1}^{N}[y_k - F_\lambda^{[k]}(\mathbf{x}_k)]^2 \tag{5.113}$$

$V_0(\lambda)$ 仅依赖于数据点本身。这样 λ 的普通交叉确认估计即为使 $V_0(\lambda)$ 最小化的函数(Wahba, 1990)。

$F_\lambda^{[k]}(\mathbf{x}_k)$ 一个有用的性质是如果用预测 $F_\lambda^{[k]}(\mathbf{x}_k)$ 来代替数据点 y_k 的值，使用数据点 $y_1, y_2, \cdots, y_{k-1}, y_k, y_{k+1}, \cdots, y_N$ 使(5.98)的原始 Tikhonov 泛函 $\mathscr{E}(F)$ 最小，则 $F_\lambda^{[k]}(\mathbf{x}_k)$ 就是所求的解。这个性质以及对于每一个输入向量 $\mathbf{x}$，$\mathscr{E}(F)$ 的最小化函数 $F_\lambda(\mathbf{x})$ 线性依赖于 y_k，这使我们有

$$F_\lambda^{[k]}(\mathbf{x}_k) = F_\lambda(\mathbf{x}_k) + (F_\lambda^{[k]}(\mathbf{x}_k) - y_k)\frac{\partial F_\lambda(\mathbf{x}_k)}{\partial y_k} \tag{5.114}$$

由式(5.100)所定义的影响矩阵 $\mathbf{A}(\lambda)$ 的分量，我们很容易看出

$$\frac{\partial F_\lambda(\mathbf{x}_k)}{\partial y_k} = a_{kk}(\lambda) \tag{5.115}$$

其中 $a_{kk}(\lambda)$ 是影响矩阵 $\mathbf{A}(\lambda)$ 对角线上的第 k 个元素。将式(5.115)代入式(5.114)中并解 $F_\lambda^{[k]}(\mathbf{x}_k)$ 的方程，可得

$$F_\lambda^{[k]}(\mathbf{x}_k) = \frac{F_\lambda(\mathbf{x}_k) - a_{kk}(\lambda)y_k}{1 - a_{kk}(\lambda)} = \frac{F_\lambda(\mathbf{x}_k) - y_k}{1 - a_{kk}(\lambda)} + y_k \tag{5.116}$$

将式(5.116)代入式(5.113)中，我们就可重新定义 $V_0(\lambda)$ 为

288

$$V_0(\lambda) = \frac{1}{N}\sum_{k=1}^{N}\left[\frac{y_k - F_\lambda(\mathbf{x}_k)}{1 - a_{kk}(\lambda)}\right]^2 \tag{5.117}$$

但是，对于不同的 k，$a_{kk}(\lambda)$ 的值是不同的，这说明不同的数据点在 $V_0(\lambda)$ 中具有不同的作用。为了避免通常的交叉确认的这一特性，Craven and Wahba(1979)通过坐标旋转[1]引入了广义交叉确认(generalized cross-validation, GCV)。特别地，式(5.117)中的 $V_0(\lambda)$ 改变为

$$V(\lambda) = \frac{1}{N}\sum_{k=1}^{N}\omega_k\left[\frac{y_k - F_\lambda(\mathbf{x}_k)}{1 - a_{kk}(\lambda)}\right]^2 \tag{5.118}$$

其中，权系数 ω_k 由

$$\omega_k = \left[\frac{1 - a_{kk}(\lambda)}{\frac{1}{N}\mathrm{tr}[\mathbf{I} - \mathbf{A}(\lambda)]}\right]^2 \tag{5.119}$$

定义。这样广义交叉确认函数 $V(\lambda)$ 就变为

$$V(\lambda) = \frac{\frac{1}{N}\sum_{k=1}^{N}[y_k - F_\lambda(\mathbf{x}_k)]^2}{\left[\frac{1}{N}\mathrm{tr}[\mathbf{I} - \mathbf{A}(\lambda)]\right]^2} \tag{5.120}$$

最后，将式(5.100)代入式(5.120)，可得

$$V(\lambda) = \frac{\frac{1}{N}\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{\left[\frac{1}{N}\mathrm{tr}[\mathbf{I} - \mathbf{A}(\lambda)]\right]^2} \tag{5.121}$$

上式在计算上仅依赖于和数据有关的量。

广义交叉确认函数 $V(\lambda)$的最优性

令 λ 表示广义交叉确认函数 $V(\lambda)$期望值的最小化函数。广义交叉确认的期望无效度可定义为

$$I^* = \frac{E[R(\lambda)]}{\min_\lambda E[R(\lambda)]} \tag{5.122}$$

其中 $R(\lambda)$是由式(5.99)定义的数据集的均方误差。自然，I^* 的渐进值满足条件

$$\lim_{N\to\infty} I^* = 1 \tag{5.123}$$

289

换句话说，对于一个很大的 N，使 $V(\lambda)$最小的 λ，同时也使 $R(\lambda)$接近最小的可能值，这使得 $V(\lambda)$成为一个很好的估计 λ 的工具。

评论小结

一般的想法是选择一个使在整个数据集上的均方差 $R(\lambda)$最小化的 λ 值。但是这一想法不能直接实现，因为 $R(\lambda)$中包含有未知的回归函数 $f(\mathbf{x})$。因此，在实际中我们就要分两种可能性来处理：

- 如果噪声方差 σ^2 已知，我们就选择使式(5.110)的估计 $\hat{R}(\lambda)$最小化的 λ 作为最佳值，这里所谓的最佳是指它也使 $R(\lambda)$最小化。
- 如果 σ^2 未知，我们可以选择使得式(5.121)的广义交叉确认函数 $V(\lambda)$最小化的 λ 作为好的选择，当 $N\to\infty$时，这个 λ 可以使期望均方误差逼近其最小可能值。

值得注意的是，使用广义交叉确认方法估计 λ 所依赖的理论是渐近的。只有当所得的数据集大到能使信号和噪声相分离的程度，这种方法才能希望得到令人满意的结果。

在实际使用中，广义交叉确认方法对于非齐次方差和非 Gauss 噪声情况，表现出很强的鲁棒性(Wahba,1990)。但是如果噪声过程是高度相关的，这种方法往往得不到满意的正则化参数 λ 的估计。

最后需要说明的是广义交叉确认函数的计算问题。对于一个给定的正则化参数的试验值 λ，求式(5.121)中分母$[\mathrm{tr}[\mathbf{I}-\mathbf{A}(\lambda)]/N]^2$ 将是计算 $V(\lambda)$中计算量最大的部分。在 Wahba et al.(1995)中描述的“随机化迹方法”可以用于计算 $\mathrm{tr}[\mathbf{A}(\lambda)]$；这种方法可用于超大规模的系统。

5.10 RBF 网络的逼近性质

第 4 章讨论了多层感知器的逼近性质。与多层感知器类似，径向基函数也具有优良的逼近特性。RBF 网络族足够大，它可以在一个紧集上一致逼近任何连续函数[12]。

通用逼近定理

令 G：$\mathbb{R}^{m_0}\to\mathbb{R}$ 是一个可积的有界连续函数，且满足

$$\int_{\mathbb{R}^{m_0}} G(\mathbf{x})\,d\mathbf{x} \neq 0$$

290

令$\mathcal{G}_G$ 表示一个 RBF 网络族，它由函数 F：$\mathbb{R}^{m_0}\to\mathbb{R}$ 组成，其中

$$F(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G\left(\frac{\mathbf{x}-\mathbf{t}_i}{\sigma}\right)$$

上式中 $\sigma>0$，对所有的 $i=1, 2, \cdots, m_1$ 有 $w_i \in \mathbb{R}$ 且 $\mathbf{t}_i \in \mathbb{R}^{m_0}$。这样，我们就可叙述 RBF 网络的通用逼近定理如下(Park and Sandberg, 1991)：

对任何输入－输出映射函数 $f(\mathbf{x})$，存在一个 RBF 网络，其中心集合为 $\{\mathbf{t}_i\}_{i=1}^{m_1}$，公共宽度为 $\sigma>0$，使得由该 RBF 网络实现的输入输出映射函数 $F(\mathbf{x})$ 在 $L_p(p\in[1,\infty])$ 范数下接近于 $f(\mathbf{x})$。

注意在通用逼近定理中，并不要求核 $G: \mathbb{R}^{m_0}\to\mathbb{R}$ 具有径向对称性，因此该定理强于 RBF 网络的必要性。最重要的是该定理在实际应用中为使用径向基函数设计神经网络提供了理论基础。

维数灾(再讨论)

除了 RBF 网络的通用逼近性质外，我们还必须考虑这种神经网络所能达到的逼近率。在第4章的讨论中，我们知道一类逼近函数的固有复杂度与比率 m_0/s 成指数增长关系，其中 m_0 是输入维数(即输入空间的维数)，s 是光滑度指数(度量加在特定逼近函数类中的逼近函数上的约束数目)。Bellman 的维数灾理论告诉我们，不管你采用什么样的逼近技术，如果光滑度指数 s 维持常数，则达到具有某一规定的精确度的逼近函数所需的参数数与输入维数 m_0 成指数增长关系。要想达到某一与输入维数 m_0 无关收敛率，从而避免维数灾，惟一的办法就是使光滑度指数 s 与逼近函数的参数数目一起增长使其弥补复杂度的增加。这一点在表5-3说明，摘自 Girosi and Anzellotti(1992)。表5-3总结想要得到独立于输入维数 m_0 的收敛率，用多层感知器逼近技术及 RBF 网络逼近技术时，其函数空间所应满足的约束。当然，加于这两种逼近技术的约束各不相同，这反映它们所遵循的实现公式的不同。在 RBF 网络情形，结果在 Sobolev 函数空间[13]成立，其中的函数直到 $2m>m_0$ 阶的导数是可积的。换句话说，要求逼近函数导数可积的阶数随着输入维数 m_0 的增加而增加，以使收敛率与 m_0 无关。如第4章中的解释，多层感知器模型有相似的约束，但以相当隐晦的方式。从表5-3得到的结论可陈述如下：

在多层感知器和 RBF 网络中可实现的逼近函数空间中，随着输入维数 m_0 的增加，空间的约束也将增加。

最后的结果是无论使用多层感知器或 RBF 网络的神经网络技术还是使用其他具有类似特性的非线性技术都不可能打破维数灾。

表5-3 具有相同的收敛率 $O(1/\sqrt{m_1})$ 的两个逼近技术和它们相应的函数空间，其中 m_1 为隐藏空间的大小

函数空间	范 数	逼近技术
$\int_{\mathbb{R}^{m_0}} \|\mathbf{s}\| \tilde{F}(\mathbf{s})d\mathbf{s}<\infty$ 其中 $\tilde{F}(\mathbf{s})$ 为逼近函数 $F(\mathbf{x})$ 的 Fourier 变换	$L_2(\Omega)$	(a)多层感知器：$F(\mathbf{x})=\sum_{i=1}^{m_1} a_i\varphi(\mathbf{w}_i^T\mathbf{x}+b_i)$ 其中 $\varphi(\cdot)$ 为 sigmoid 激活函数
具有直到 $2m>m_0$ 阶可积导数的函数组成的 Sobolev 空间	$L_2(\mathbb{R}^2)$	(b)RBF 网络：$F(\mathbf{x})=\sum_{i=1}^{m_1} a_i\exp\left(-\frac{\|\mathbf{x}-\mathbf{t}_i\|^2}{2\sigma^2}\right)$

291

样本复杂性、计算复杂性及泛化能力的关系

实际上我们所拥有的数据量是有限的而不是无限的；在讨论中如果不考虑到这一点，那么关于逼近问题的讨论是不完全的。同样地，我们所建立的神经网络其计算复杂性也是有限的，而不是无限的。所以，如第2章所讨论的，对于在一个有限的已知样本数据上训练和在以前未遇到的数据上测试的神经网络，其泛化误差包括两部分。一部分称为逼近误差，来源于神经网络表示一个目标函数的能力是有限的。另一部分我们称之为估计误差，它来源于训练样本中所包含的目标函数的信息是有限的。使用这样的分解，Niyogi and Girosi(1996)推导出用隐藏层大小及训练样本大小表示的Gauss型RBF网络的泛化误差的界。他们推导的结果是针对用式(5.95)所描述的一种模型学习一个属于某个Sobolev空间的回归函数的情况。

这个界使用第2章描述的PAC学习的术语可叙述如下(Niyogi and Girosi,1996)：

令G表示具有m_0个输入(源)节点和m_1个隐藏单元的一类Gauss型RBF网络。令$f(\mathbf{x})$表示属于某个Sobolev空间的回归函数。假设训练样本$\mathcal{T}=\{(\mathbf{x}_i,d_i)\}_{i=1}^{N}$是基于$f(\mathbf{x})$的回归模型而随机抽取得到的。那么，对于任一置信参数$\delta\in(0,1]$，由网络产生的泛化误差的上界为

$$O\left(\frac{1}{m_1}\right)+O\left(\frac{m_0 m_1}{N}\log(m_1 N)+\frac{1}{N}\log\left(\frac{1}{\delta}\right)^{1/2}\right) \qquad (5.124)$$

292

的概率大于$1-\delta$。

由式(5.124)可得以下推论：

- 只有当隐藏单元的个数m_1增长的速度远比训练样本的大小N的增长速度慢时，泛化误差才能趋向于零。
- 对于给定的训练样本数N，隐藏单元的最佳数目m_1^*具有如下性质(参见习题5.11)

$$m_1^* \propto N^{1/3} \qquad (5.125)$$

- RBF网络所展现的收敛率$O(1/m_1)$与Barron(1993)导出的以sigmoid函数作为激活函数的多层感知器的收敛率是类似的；参看4.12节的讨论。

5.11 RBF网络与多层感知器的比较

径向基函数(RBF)网络与多层感知器(MLP)都是非线性的层状前馈网络的例子。它们都是通用逼近器。所以，毫不奇怪对于一个特定的MLP总存在一个RBF网络能够精确的模仿它，反之亦然。然而，这两种网络在几个重要方面有存在着不同之处。

1. 一个RBF网络(在其最基本的形式中)只具有一个隐藏层，而一个MLP却可以有一个或者多个隐藏层。

2. 典型地，一个MLP位于隐藏层或输出层的计算结点，其神经元模型是相同的。而RBF网络隐藏层中计算节点与网络输出层中节点是相当不同且作用也不一样。

3. RBF网络的隐藏层是非线性的，而输出层是线性的。但是MLP作为模式分类器，其隐层和输出层都是非线性的。当MLP用于解决非线性回归问题时，线性输出层通常是好的选择。

4. RBF网络每一隐藏单元的激活函数的自变量都要计算输入向量和该单元的中心之间的

Euclid 范数(距离)。同时，MLP 隐藏单元的激活函数却只要计算输入向量和与该隐藏单元相关的权值向量的内积。

5.MLP 建立一个输入 - 输出映射的全局逼近。另一方面，RBF 网络却是以指数衰减的局部非线性(如 Gauss 函数)来局部逼近一个非线性输入 - 输出映射。

结果这意味着当逼近一个非线性的输入 - 输出映射时，在相同的精度要求下，MLP 需要的参数数比 RBF 网络所需要的参数数少。

RBF 网络输出单元的线性特性说明这样的网络与 Rosenblatt 感知器的联系比和与多层感知器的联系更紧密。然而，RBF 网络与感知器是不同的，因为它能实现对输入空间进行任意的非线性变换。这一点在 XOR 问题上已经说明得很清楚，因为 XOR 问题不能用任何线性感知器来解决，但能由 RBF 网络来解决。

293

5.12 核回归及其与 RBF 网络的关系

目前为止给出的 RBF 网络的理论都是建立在插值的概念上的。在这一节中，我们将采用另一种观点，即建立在密度估计的概念之上的*核回归*(kernel regression)的观点。

具体地，再次考虑式(5.95)的回归模型，为了方便表达将其重写在下面：

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1,2,\cdots,N$$

我们可以取某一点 $\mathbf{x}$ 附近的观测值(即模型输出 y 的值)的均值作为未知回归函数 $f(\mathbf{x})$的合理估计。但是，为了达到这一目标，局部平均必须限制在 $\mathbf{x}$ 的一个较小的邻域内(即接受域)，因为一般情况下，离 $\mathbf{x}$ 较远的区域将会有不同的观察值。进一步，从第 2 章给出的讨论我们得到 $f(\mathbf{x})$等于给定 $\mathbf{x}$ 条件下 y 的条件均值(即在 $\mathbf{x}$ 上 y 的回归)，表示为

$$f(\mathbf{x}) = E[y \mid \mathbf{x}]$$

利用随机变量的期望公式，我们有

$$f(\mathbf{x}) = \int_{-\infty}^{\infty} y f_Y(y \mid \mathbf{x})\,dy \tag{5.126}$$

其中 $f_Y(y|\mathbf{x})$是 Y 以 $\mathbf{x}$ 为条件的条件概率密度函数(probability density function, pdf)。由概率论，我们有

$$f_Y(y \mid \mathbf{x}) = \frac{f_{\mathbf{X},Y}(\mathbf{x},y)}{f_{\mathbf{X}}(\mathbf{x})} \tag{5.127}$$

上式中 $f_{\mathbf{X}}(\mathbf{x})$是 $\mathbf{x}$ 的 pdf，$f_{\mathbf{X},Y}(\mathbf{x},y)$是 $\mathbf{X}$ 和 Y 的联合 pdf。因此，将式(5.127)代入式(5.126)，我们得到回归函数的下列公式：

$$f(\mathbf{x}) = \frac{\int_{-\infty}^{\infty} y f_{\mathbf{X},Y}(\mathbf{x},y)\,dy}{f_{\mathbf{X}}(\mathbf{x})} \tag{5.128}$$

294

我们感兴趣的是联合概率密度函数 $f_{\mathbf{X},Y}(\mathbf{x},y)$未知的情况。我们所知的只有训练样本$\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$。为了估计 $f_{\mathbf{X},Y}(\mathbf{x},y)$以及 $f_{\mathbf{X}}(\mathbf{x})$，可以应用一个非参数估计器，通称为 Parzen-Rosenblatt *密度估计器*(Rosenblatt, 1956, 1970; Parzen, 1962)。形成该估计器的基础是核，用符号 $K(\mathbf{x})$表示，它具有与概率密度函数相同的性质：

- *核 $K(\mathbf{x})$是一个关于 $\mathbf{x}$ 的连续有界的实函数，它关于原点对称，且在原点取得最大值。*

- 在核 $K(\mathbf{x})$ 的曲面下的总体积等于1；即对于一个 m 维的向量 $\mathbf{x}$ 有

$$\int_{\mathbb{R}^m} K(\mathbf{x})\,d\mathbf{x} = 1 \tag{5.129}$$

假设 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$ 是独立同分布的随机向量，我们可以定义 $f_{\mathbf{X}}(\mathbf{x})$ 的 Parzen-Rosenblatt 密度估计为

$$\hat{f}_{\mathbf{X}}(\mathbf{x}) = \frac{1}{Nh^{m_0}}\sum_{i=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \qquad \mathbf{x}\in\mathbb{R}^{m_0} \tag{5.130}$$

其中光滑度参数 h 是正数，称为带宽或简称为宽；h 控制着核的宽度。（请注意不要将这里的 h 与5.5节中定义 Fréchet 导数的 h 相混淆。）Parzen-Rosenblatt 密度估计器的一个重要性质是它是相容估计器[14]（即渐进无偏的），意味着如果选择 $h = h(N)$ 为 N 的函数使得

$$\lim_{N\to\infty} h(N) = 0$$

那么

$$\lim_{N\to\infty} E[\hat{f}_{\mathbf{X}}(\mathbf{x})] = f_{\mathbf{X}}(\mathbf{x})$$

为了上式成立，$\mathbf{x}$ 必须是 $\hat{f}_{\mathbf{X}}(\mathbf{x})$ 中的连续点。

用与式(5.130)描述的类似方法，我们可以得到联合概率密度函数 $f_{\mathbf{X},Y}(\mathbf{x},y)$ 的 Parzen-Rosenblatt 密度估计如下：

$$\hat{f}_{\mathbf{X},Y}(\mathbf{x},y) = \frac{1}{Nh^{m_0+1}}\sum_{i=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)K\left(\frac{y-y_i}{h}\right),\ \mathbf{x}\in\mathbb{R}^{m_0},\ y\in\mathbb{R} \tag{5.131}$$

对 $\hat{f}_{\mathbf{X},Y}(\mathbf{x},\ y)$ 作关于 y 的积分，可得式(5.130)的 $\hat{f}_{\mathbf{X}}(\mathbf{x})$，且我们应该如此。而且

$$\int_{-\infty}^{\infty} y\hat{f}_{\mathbf{X},Y}(\mathbf{x},y)\,dy = \frac{1}{Nh^{m_0+1}}\sum_{i=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)\int_{-\infty}^{\infty} yK\left(\frac{y-y_i}{h}\right)dy$$

对上式作变量代换，令 $z = (y-y_i)/h$，再利用核 $K(\cdot)$ 的对称性可得

$$\int_{-\infty}^{\infty} y\hat{f}_{\mathbf{X},Y}(\mathbf{x},y)\,dy = \frac{1}{Nh^{m_0}}\sum_{i=1}^{N} y_i K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \tag{5.132}$$

295

因此，将式(5.132)和(5.130)分别作为式(5.128)的分子和分母的估计，消去相同项后，我们可得回归函数 $f(\mathbf{x})$ 的下列估计：

$$\mathbf{F}(\mathbf{x}) = \hat{f}(\mathbf{x}) = \frac{\sum_{i=1}^{N} y_i K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)}{\sum_{j=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_j}{h}\right)} \tag{5.133}$$

为了清晰起见，上式中我们将分母中的求和下标 i 换为 j。就像一般的 RBF 网络，由式(5.133)定义的核回归估计器 $F(\mathbf{x})$ 是一个通用逼近器。

我们可以有两种方式来分析逼近函数 $F(\mathbf{x})$：

1. Nadaraya-Watson 回归估计器。定义归一化加权函数

$$W_{N,i}(\mathbf{x}) = \frac{K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)}{\sum_{j=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_j}{h}\right)}, \qquad i = 1,2,\cdots,N \tag{5.134}$$

其中

$$\sum_{i=1}^{N} W_{N,i}(\mathbf{x}) = 1,\text{对所有的 }\mathbf{x} \tag{5.135}$$

我们可将式(5.133)所示的核回归估计简写成

$$F(\mathbf{x}) = \sum_{i=1}^{N} W_{N,i}(\mathbf{x}) y_i \tag{5.136}$$

它将 $F(\mathbf{x})$ 描述为观察值 y 的加权均值。式(5.136)给出的加权函数 $W_{N,i}(\mathbf{x})$ 形式是由 Nadaraya(1964)和 Watson(1964)提出的，所以式(5.136)所示的逼近函数称为 Nadaraya-Watson 回归估计器(Nadaraya-Watson regression estimator, NWRE)[15]。

2. 归一化的 RBF 网络。对于第二种观点，我们假设核 $K(\mathbf{x})$ 是球对称的，这样我们就可以令(Kržyzak et al.,1996)

$$K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) = K\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right) \qquad \text{对所有 } i \tag{5.137}$$

这里 $\|\cdot\|$ 表示包含向量的欧几里德范数。相应地我们定义归一化径向基函数为

[296]

$$\Psi_N(\mathbf{x},\mathbf{x}_i) = \frac{K\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right)}{\sum_{j=1}^{N} K\left(\frac{\|\mathbf{x}-\mathbf{x}_j\|}{h}\right)}, \qquad i = 1,2,\cdots,N \tag{5.138}$$

其中，对所有的 $\mathbf{x}$ 有

$$\sum_{i=1}^{N} \Psi_N(\mathbf{x},\mathbf{x}_i) = 1 \tag{5.139}$$

$\Psi_N(\mathbf{x},\mathbf{x}_i)$ 中的下标 N 表示使用归一化(normalization)。

对于这里所讨论的回归问题，我们可以看出应用于基函数 $\Psi_N(\mathbf{x}, \mathbf{x}_i)$ 的“线性权值” w_i，就是回归模型中对应于 $\mathbf{x}_i$ 的观察值 y_i。因此令

$$y_i = w_i, \qquad i = 1,2,\cdots,N$$

我们可以重新将式(5.133)所示的逼近函数写成一般形式

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i \Psi_N(\mathbf{x},\mathbf{x}_i) \tag{5.140}$$

式(5.140)表示的是一个归一化 RBF 网络的输入－输出映射(Moody and Darken, 1989; Xu et al., 1994)。注意，对所有的 $\mathbf{x}$ 和 $\mathbf{x}_i$

$$0 \leq \Psi_N(\mathbf{x},\mathbf{x}_i) \leq 1 \tag{5.141}$$

因此，$\Psi_N(\mathbf{x},\mathbf{x}_i)$ 可以解释为以 $\mathbf{x}_i$ 为条件的由输入向量 $\mathbf{x}$ 描述的事件的概率。

式(5.138)的归一化径向基函数 $\Psi_N(\mathbf{x},\mathbf{x}_i)$ 与一般径向基函数的不同之处在于 $\Psi_N(\mathbf{x},\mathbf{x}_i)$ 有一个组成归一化因子的分母。归一化因子是关于输入向量 $\mathbf{x}$ 的固有 pdf。因此，对所有的 $\mathbf{x}$ 基函数 $\Psi_N(\mathbf{x},\mathbf{x}_i)$ 的 N 项之和等于 1，即式(5.139)。与此相对，一般 RBF 网络的基(格林)函数(式 5.57)却不一定满足这个条件。

这里关于式(5.138)描述的输入－输出映射 $F(\mathbf{x})$ 的推导应用了密度估计的概念。与超曲面的重建问题相似，密度估计是一个不适定的问题。为了使其适定，必须应用正则化的某种形式。我们可以在正则化理论(Vapnik, 1982)的框架下推导 Parzen-Rosenblatt 密度估计器，从而推导 Nadaraya-Watson 回归估计器。当然，密度估计中的代价泛函与式(5.23)的确定性 Tikhonov 泛函有所不同。密度估计中的代价泛函包括两项：一个包含未知概率密度函数的误差平方项和一个稳定泛函的适当形式。

多元 Gauss 分布

一般说来可以选择各种各样的核函数。但是，理论和实际的考虑限制了我们的选择。与格林函数一样，广泛地使用多元 Gauss 分布作为核函数：

$$K(\mathbf{x}) = \frac{1}{(2\pi)^{m_0/2}}\exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right) \tag{5.142}$$

[297]

其中，m_0 是输入向量 $\mathbf{x}$ 的维数。很明显，式(5.142)所示的核 $K(\mathbf{x})$ 具有球对称性。假设使用相同的宽度(扩展)σ，σ 与光滑参数 h 对每一个 Gauss 分布的作用相同，且以 $\mathbf{x}_i$ 作为核函数的中心，我们可写成

$$K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) = \frac{1}{(2\pi\sigma^2)^{m_0/2}}\exp\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right), i = 1,2,\cdots,N \tag{5.143}$$

因此，使用式(5.143)，Nadaraya-Watson 回归估计可以写成(Specht，1991)

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{N} y_i \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{j=1}^{N}\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_j\|^2}{2\sigma^2}\right)} \tag{5.144}$$

其中分母项表示 Parzen-Rosenblatt 密度估计器，由 N 个以数据点 $\mathbf{x}_1$，$\mathbf{x}_2$，…，$\mathbf{x}_N$ 为中心的多元 Gauss 分布之和构成。

相应地，将式(5.143)代入(5.138)和(5.140)，可以得到归一化 RBF 网络的输入－输出映射函数的如下形式：

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{N} w_i \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{j=1}^{N}\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_j\|^2}{2\sigma^2}\right)} \tag{5.145}$$

在式(5.144)和式(5.145)中，归一化径向基函数的中心与输入数据点 $\{\mathbf{x}_i\}_{i=1}^{N}$ 一致。与一般径向基函数相同，可以使用较小数量的归一化径向基函数，它们的中心看作自由参数可以根据某种启发式方法选择，或者可以按某种原则确定(Poggio and Girosi,1990a)。

5.13 学习策略

如果不考虑其理论背景，对径向基函数(RBF)网络采取的学习过程可作如下分析。与网络输出单元相连的线性权值与隐藏单元的非线性激活函数相比是在一种不同的"时间尺度"上更新的。因此，当隐藏层的激活函数根据某种非线性最优策略进行缓慢更新的时候，输出权值却是根据线性最优策略进行快速调整。重要的是，在 RBF 网络中，不同的层起着不同的作用。因此，对于隐藏层和输出层采用不同的最优策略是合理的，也许可以使用不同的时间尺度来实现(Lowe,1991a)。

[298]

根据网络径向基函数中心的确定方法不同，在设计 RBF 网络上有不同的学习策略。这里我们将介绍四种方法。前三种设计策略是建立在插值理论的基础之上的。最后一种设计策略将结合正则化理论和核回归估计理论的理论。

1. 随机选取固定中心

最简单的方法是假设定义隐藏单元的激活函数是固定径向基函数。中心的位置可以用随机的方式从训练数据集合中选取。如果训练数据是以当前问题的典型方式分布的，则该方法可以被认为是一个"明智"的方法(Lowe,1989)。对于径向基函数本身，我们可以用一个各向同性的 Gauss 函数，它的标准偏差是根据中心的散布而固定的。特别地，一个以 $\mathbf{t}_i$ 为中心的(归一化的)径向基函数定义为

$$G(\|\mathbf{x}-\mathbf{t}_i\|^2)=\exp\left(-\frac{m_1}{d_{max}^2}\|\mathbf{x}-\mathbf{t}_i\|^2\right),\qquad i=1,2,\cdots,m_1 \tag{5.146}$$

其中 m_1 是中心的数目，d_{max}是所选中心之间的最大距离。可以看出，所有 Gauss 径向基函数的标准偏差(即宽度)都固定为

$$\sigma=\frac{d_{max}}{\sqrt{2m_1}} \tag{5.147}$$

上式保证每一个径向基函数都不会太尖，也不会太平；这两种极端情况都应该尽量避免。作为(5.147)的另一种选择，我们也可以在数据密度较低的区域上使用个别放大的宽度较大的中心，这要求对训练数据作实验。

在这种方法中，惟一需要学习的参数就是输出层上的线性权值。求输出权值的一个直接的方法就是伪逆法(Broomhead and Lowe,1988)。特别地，我们有(也可参看式(5.77)和式(5.78))

$$\mathbf{w}=\mathbf{G}^+\mathbf{d} \tag{5.148}$$

其中 $\mathbf{d}$ 是训练集合中的期望响应向量。矩阵 $\mathbf{G}^+$ 是矩阵 $\mathbf{G}$ 的伪逆，而矩阵 $\mathbf{G}$ 定义为

$$\mathbf{G}=\{g_{ji}\} \tag{5.149}$$

其中

299

$$g_{ji}=\exp\left(-\frac{m_1}{d^2}\|\mathbf{x}_j-\mathbf{t}_i\|^2\right),j=1,2,\cdots,N;i=1,2,\cdots,m_1 \tag{5.150}$$

上式中 $\mathbf{x}_j$ 是训练样本中第 j 个输入向量。

求一个矩阵的伪逆的所有计算的基础是奇异值分解(SVD)(Golub and Van Loan,1996)：

如果 $\mathbf{G}$ 是一个 $N\times M$ 阶的实矩阵，则存在正交矩阵

$$\mathbf{U}=\{\mathbf{u}_1,\mathbf{u}_2,\cdots,\mathbf{u}_N\}$$

和

$$\mathbf{V}=\{\mathbf{v}_1,\mathbf{v}_2,\cdots,\mathbf{v}_M\}$$

使得

$$\mathbf{U}^T\mathbf{G}\mathbf{V}=\mathrm{diag}(\sigma_1,\sigma_2,\cdots,\sigma_K),K=\min(M,N) \tag{5.151}$$

其中

$$\sigma_1\geqslant\sigma_2\geqslant\cdots\geqslant\sigma_K>0$$

矩阵 $\mathbf{U}$ 的列向量称为 $\mathbf{G}$ *左奇异向量*，矩阵 $\mathbf{V}$ 的列向量称为 $\mathbf{G}$ *右奇异向量*。$\sigma_1,\sigma_2,\cdots,\sigma_K$ 称为*奇异值*。根据奇异值分解定理，矩阵 $\mathbf{G}$ 的 $M\times N$ 阶伪逆定义为

$$\mathbf{G}^+=\mathbf{V}\boldsymbol{\Sigma}^+\mathbf{U}^T \tag{5.152}$$

其中 $\boldsymbol{\Sigma}^+$ 是一个由 $\mathbf{G}$ 的奇异值决定的 $N\times N$ 阶矩阵，

$$\boldsymbol{\Sigma}^+=\mathrm{diag}\left(\frac{1}{\sigma_1},\frac{1}{\sigma_2},\cdots,\frac{1}{\sigma_K},0,\cdots,0\right) \tag{5.153}$$

计算矩阵伪逆矩阵的有效算法在 Golub and Van Loan(1996)中有详细介绍。

有趣的是，根据应用随机选取中心方法的经验表明，这种方法相对来说对正则化的使用不太敏感；参看习题 5.14，它使用这种方法在计算机上实现模式分类。这种性能提示，从一个固定大小的大规模训练集合中随机选取中心的 RBF 网络设计方法，就其自身而言也许就是一种正则化的方法。

2. 中心的自组织选择

刚才描述的固定中心的方法主要缺陷是为了达到性能的满意水平需要一个巨大的训练集合。克服这一限制的一个方法就是使用一种混合学习过程，包括下面两个不同的阶段(Moody and Darken,1989;Lippmann,1989b;Chen et al.,1992)：

- *自组织学习阶段*，它的目的是为隐藏层径向基函数的中心估计一个合适的位置。
- *监督学习阶段*，它通过估计输出层的权值完成神经网络的设计。

300

虽然可以用批处理来执行上述两种学习阶段，但是用自适应(迭代)的方法更理想。

对于自组织学习过程，我们需要一个聚类的算法将所给的数据点剖分成几个不同的部分，每一部分中的数据都尽量有相同的性质。一种这样的算法为 *k-均值聚类算法*(Duda and Hart,1973)，它将径向基函数的中心放在输入空间$\mathscr{X}$中重要数据点所在的区域上。令 m_1 表示径向基函数数目；m_1 要依靠试验来决定取何种适合值。令$\{\mathbf{t}_k(n)\}_{k=1}^{m_1}$表示径向基函数在第 n 次迭代时的中心。那么，k-均值聚类算法进行如下：

1. *初始化*。选择随机值作为中心 $\mathbf{t}_k(0)$的初始值；惟一限制是要求每一个中心的初值不同。将中心的欧几里德范数保持为较小的值可能会更理想一些。

2. *抽取样本*。在输入空间$\mathscr{X}$中以某种概率抽取样本向量 $\mathbf{x}$，作为第 n 次迭代的输入向量。

3. *相似匹配*。令 $k(\mathbf{x})$表示输入向量 $\mathbf{x}$ 的最佳匹配(竞争获胜)中心的下标值。第 n 次迭代时按欧几里德最小距离准则确定 $k(\mathbf{x})$的值：

$$k(\mathbf{x}) = \arg\min_k \|\mathbf{x}(n) - \mathbf{t}_k(n)\|, k = 1,2,\cdots,m_1 \tag{5.154}$$

其中 $\mathbf{t}_k(n)$表示第 k 个径向基函数在第 n 次迭代时的中心。

4. *更新*。用下述规则调整径向基函数的中心：

$$\mathbf{t}_k(n+1) = \begin{cases} \mathbf{t}_k(n) + \eta[\mathbf{x}(n) - \mathbf{t}_k(n)], & k = k(\mathbf{x}) \\ \mathbf{t}_k(n), & \text{其他情况} \end{cases} \tag{5.155}$$

其中 η 是学习率，且 $0<\eta<1$。

5. *继续*。将 n 的值加 1，回到第 2 步，重复上述过程，直到中心 $\mathbf{t}_k$ 的改变量很小时为止。

这里所说的 k-均值聚类算法实际上是竞争(胜者全得)学习过程的一种特殊情况，它通称为*自组织映射*，我们将在第 9 章中详细讨论。后一算法也适于实现自组织学习阶段。

k-均值聚类算法的一个局限在于它只能达到依赖于所选中心初值的局部最优解。因此，计算资源就有可能浪费，因为一些中心的初值可能位于输入空间中稀少数据点的区域，因此它们没有机会移到它们所需的新位置去。最终的结果可能就会是不必要的大网络。为了克服传统的 k-均值聚类算法的局限，Chen(1995)提出了使用一种*增强 k-均值聚类算法*，

该算法归功于 Chinrunrueng and Séquin(1994)，它建立在变差加权度量的聚类基础上，可以使算法收敛于一个最优结果或者近似最优结果，而与中心的初始位置无关。

在利用 k – 均值聚类算法或者它的增强形式得到每一个 Gauss 型径向基函数的中心及其宽度后，混合学习过程余下的最后一步是估计输出层的权值。一个最简单的估计方法就是在 301 第3章中介绍过的最小均方(LMS)算法。隐藏单元产生的输出信号向量构成 LMS 算法的输入向量。注意，应用于隐藏单元的 k – 均值聚类算法和应用于输出单元的 LMS 算法可以用并行的方式分别进行各自的计算，从而加快训练过程。

3. 中心的监督选择

在第3种方法中，径向基函数的中心以及网络的所有其他自由参数都将经历一个监督学习的过程。换句话说，RBF 网络将采取其最一般的方式。这个方法的自然后选是采用误差修正学习过程，这种方法可以很方便地采用梯度下降法，它代表 LMS 算法的一种推广。

建立这种学习过程的第一步是定义代价函数的瞬时值

$$\mathscr{E} = \frac{1}{2}\sum_{j=1}^{N} e_j^2 \tag{5.156}$$

其中 N 是用于学习的训练样本数目，e_j 是误差信号，定义如下：

$$e_j = d_j - F^*(\mathbf{x}_j) = d_j - \sum_{i=1}^{M} w_i G(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i}) \tag{5.157}$$

目标是找到使$\mathscr{E}$最小的自由参数 w_i，$\mathbf{t}_i$ 和 $\mathbf{\Sigma}_i^{-1}$ 的值(后者和范数加权矩阵 $\mathbf{C}_i$ 有关)。最小化的结果列于表5-4中，这些结果的推导将在习题5.13中作为练习留给读者。表5-4中有几点值得注意：

表5-4　线性权值的自适应公式和 RBF 网络中心的位置和散布*

1. 线性权值(输出层)

$$\frac{\partial\mathscr{E}(n)}{\partial w_i(n)} = \sum_{j=1}^{N} e_j(n) G(\|\mathbf{x}_j - \mathbf{t}_i(n)\|_{C_i})$$

$$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial\mathscr{E}(n)}{\partial w_i(n)}, \quad i = 1,2,\cdots,m_1$$

2. 中心位置(隐藏层)

$$\frac{\partial\mathscr{E}(n)}{\partial \mathbf{t}_i(n)} = 2w_i(n)\sum_{j=1}^{N} e_j(n) G'(\|\mathbf{x}_j - \mathbf{t}_i(n)\|_{C_i})\mathbf{\Sigma}_i^{-1}[\mathbf{x}_j - \mathbf{t}_i(n)]$$

$$\mathbf{t}_i(n+1) = \mathbf{t}_i(n) - \eta_2 \frac{\partial\mathscr{E}(n)}{\partial \mathbf{t}_i(n)}, \qquad i = 1,2,\cdots,m_1$$

3. 中心扩展(隐藏层)

$$\frac{\partial\mathscr{E}(n)}{\partial \mathbf{\Sigma}_i^{-1}(n)} = -w_i(n)\sum_{j=1}^{N} e_j(n) G'(\|\mathbf{x}_j - \mathbf{t}_i(n)\|_{C_i})\mathbf{Q}_{ji}(n)$$

$$\mathbf{Q}_{ji}(n) = [\mathbf{x}_j - \mathbf{t}_i(n)][\mathbf{x}_j - \mathbf{t}_i(n)]^T$$

$$\mathbf{\Sigma}_i^{-1}(n+1) = \mathbf{\Sigma}_i^{-1}(n) - \eta_3 \frac{\partial\mathscr{E}(n)}{\partial \mathbf{\Sigma}_i^{-1}(n)}$$

* 项 $e_j(n)$是输出单元 j 在时刻 n 时的误差信号。项 $G'(\cdot)$是 Green 函数 $G(\cdot)$关于它的自变量的一阶导数。

- 代价函数$\mathscr{E}$对于线性权值 w_i 来说是凸的，但是对于中心 $\mathbf{t}_i$ 和矩阵 $\mathbf{\Sigma}_i^{-1}$ 来说却是非凸的；在后一种情况下，$\mathbf{t}_i$ 和 $\mathbf{\Sigma}_i^{-1}$ 的取值可能会陷入参数空间的上的一个局部最小值

处。

- 参数 w_i，$\mathbf{t}_i$ 和 $\mathbf{\Sigma}_i^{-1}$ 的更新公式中的学习率应为不同的值 η_1、η_2和 η_3。
- 与反向传播算法不同，表 5-4 所列的 RBF 网络的梯度下降法中没有误差反向传播。
- 梯度向量$\partial\mathscr{E}/\partial\mathbf{t}_i$ 的效果和聚类效果类似，是依赖于任务的(Poggio and Cirosi,1990a)。 [302]

在梯度下降法的初始化过程中，通常都希望由参数空间的一个结构化初始条件开始，这一条件限制搜索的参数空间区域使我们在已知的有用区域中搜索，这可以通过标准的模式分类法来实现(Lowe,1991a)。应用这一方法，收敛到权值空间非期望的局部最小值的可能性将减少。例如，我们可以从一个 Gauss 分类器开始，该分类器假设每一类中的每一个模式都是从 Gauss 分布中抽取的；基于 Bayes 假设检验过程的模式分类器的这种特殊形式在第 3 章中已经讨论过了。

在讨论的这个阶段出现的问题是：自适应选取径向基函数的中心的位置能得到什么好处？这个问题的答案当然依赖于实际应用。虽然如此，根据一些文献报告的结果，允许中心移动确实能得到一些实际的好处。Lowe(1989)将 RBF 网络应用于语音识别的工作结果表明，如果要求最小的网络配置的话，用非线性参数优化的方法是有利的。但是，据 Lowe 所言，用一个更大的 RBF 网络可以达到同样的泛化效果，这里所谓更大的神经网络就是隐藏层具有更多固定中心和仅用线性优化的方法来调整输出层的网络。

Wettschereck 和 Dietterich(1992)曾经对应用固定中心的(Gauss 型)径向基函数网络和应用可调中心的广义径向基函数网络的性能作过比较；在后一种情况中心位置是由监督学习确定的。性能比较是对 NETtalk 任务进行的。最早的 NETtalk 试验是由 Sejnowski 和 Rosenberg (1987)使用多层感知器进行的，训练所用的算法是反向传播算法；这将在第 13 章中介绍。Wettschereck 和 Dietterich 的试验目的是为了了解神经网络是如何将英语拼写映射为语音的发音。Wettschereck 和 Dietterich 在 NETtalk 上所作的试验研究可以小结如下： [303]

- RBF 网络(对中心位置采用无监督学习，对输出权值向量采用监督学习)不如多层感知器模型(采用反向传播算法)推广得好。
- 广义 RBF 网络(中心位置与输出权值均采用监督学习)的泛化能力可以明显好于多层感知器模型。

4. 正则化严格插值法

结合第 5.5 节的正则化理论和第 5.12 节的核回归估计理论的基本原理的 RBF 网络设计的方法在 Yee(1998)描述。该方法包括组合利用以下四个组成部分：

1. 径向基函数 G，可作为(可能带有某种缩放)一致(均方)Nadaraya-Watson 回归估计(NWRE)的核。

2. 对角输入范数加权矩阵 $\mathbf{\Sigma}^{-1}$，对具有

$$\mathbf{\Sigma} = \mathrm{diag}(h_1, h_2, \cdots, h_{m_0}) \tag{5.158}$$

的所有中心是共同的，其中 h_1，h_2，…，h_{m_0} 是具有(缩放后)核 G 的一致 NWRE 的每个维的带宽，如同以前设置的一样，而 m_0 是输入空间的维数。例如，我们可以设 $h_i = \alpha_i\sigma_i^2$，$i=1$，2，…，m_0，式中 σ_i^2 表示第 i 个输入变量的样本方差，它是从已知的训练输入数据中估计而来的。正的输入缩放因子 $\alpha_1, \alpha_2, \cdots, \alpha_{m_0}$ 可以通过适当的交叉确认(cross-validation, CV)过

程确定，如在5.9节解释的一样。

3. 正则化严格插值，它涉及根据式(5.54)训练线性权值。

4. 通过渐近优化的方法，例如式(5.117)所示的交替留一法或者式(5.121)所示的GCV法，选择正则化参数 λ 及缩放因子 α_1，α_2，…，α_{m_0}。选择的参数可说明如下：

- 选择的 λ 越大，则噪音对参数测量的干扰就越大。
- 当径向基函数是一个单峰值的核函数(例如Gauss型核函数)时，特定 α_i 的值越小，则整个网络的输出对相应的输入维越敏感。相反，若某个 α_i 越大，则整个网络输出对该输入维的变化就越迟钝。因此，我们可以通过 α_i 的选取来标明每一个输入变量的重要程度，从而在需要降低维数，可以将无关紧要的输入维删去。

304

上述设计过程的合理性在Yee(1998)中有详细的讨论。我们选择这种设计方法的目的可以解释如下。可以证明NWRE与一种特殊类别的正则化RBF网络相对应，也就是说，对于任意的NWRE，我们都可以构造一个适当的正则化RBF网络序列，当其正则化参数序列$\{\lambda_N\}$随着 N(训练样本的大小)趋向于无穷而(以某种恰当的速率)趋向于无穷时，RBF网络与NWRE之间的均方差和绝对误差都趋向于零。这样我们就可以用构造的RBF网络来逼近任意的NWRE。在另一方面，当 $N\to\infty$，(在某种温和的条件下)由式(5.99)所定义的风险趋向于(全局)均方误差。如果我们用渐近最优参数的方法来选取正则化参数序列，那么，通过构造，这样得到的RBF网络结果序列一定具有(渐近)最小均方差的RBF网络，这里最小是相对于所有可能的正则化参数序列的选择，其中包括与NWRE相对应的那个选择。如果已知NWRE均方误差相容的条件成立，则根据同样过程设计的RBF网络也是均方差相容的。换句话说，用上述方法得到的RBF网络继承了NWRE的相容性。由这一结论，我们可以将NWRE的相容性结果应用于诸如时间序列回归等的研究中，在这一类研究中，相关和非稳态的情况经常遇见，而假设具有独立同分布的训练数据和稳态过程的一般的神经网络对这类问题是无效的。总而言之，通过组合正则化理论和核回归估计理论的基本原理，这里列出的设计过程提供了用于正则化RBF网络设计和应用的实际规定的理论支持。

5.14 计算机实验：模式分类

在这一节中，我们将通过计算机实验来阐明基于使用严格插值法的正则化RBF网络的设计。这个计算机实验是一个二值分类问题，其中的数据是从与类$\mathscr{C}_1$和类$\mathscr{C}_2$相对应的两个等概率的交叉二维Gauss分布中抽取的。有关Gauss分布的详细内容与4.8节中所述的一样。类$\mathscr{C}_1$的均值向量为$[0,0]^T$，公共方差为1；类$\mathscr{C}_2$的均值向量为$[0,2]^T$，公共方差为4。这一节描述的计算机实验可以看作正则化RBF网络和第4.8节的反向传播学习实验的对应部分。

因为有两个类$\mathscr{C}_1$和$\mathscr{C}_2$，构造正则化RBF网络具有两个输出单元，每个对应一类。同样，二值类指示器输出用作期望输出值，表示为

$$d_k^{(p)} = \begin{cases} 1 & \text{如果 } p \text{ 属于类} \mathscr{C}_k \\ 0 & \text{其他情况} \end{cases}$$

其中 $k=1,2$。

在进行实验之前，我们必须解决确定实现模式分类的输出规则。在Yee(1998)中证明正

则化 RBF 网络分类器的输出提供一个后验类概率估计。这个结论只有在利用期望输出的二值类指示器向量类型训练网络时才成立。我们现在将式(4.55)作为这类网络的决策规则：

选择对应于最大输出函数的类.

305

中心选择的严格插值法用不同正则化参数 λ 的值进行测试。对一个指定的 λ，由式(5.54)我们就可以算出 RBF 网络输出层的权值，表示为

$$\mathbf{w} = (\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{d}$$

其中 **G** 是一个 $N \times N$ 阶的 Green 矩阵，它的第 ji 个元素是径向对称的 Green 函数 $G(\mathbf{x}_j, \mathbf{x}_i)$，$N$ 是样本的大小，**d** 是期望响应向量。

对每一个正则化参数 λ，总体由 50 个独立的网络构成，每一个网络都用具有 1000 个模式的相同的参考集进行测试。

表 5-5 给出当有 $m_1 = 20$ 个中心时正确分类概率的总体统计(ensemble statistic)。总体统计根据不同的 λ 值进行计算。表 5-6 给出的是具有 $m_1 = 100$ 个中心的 RBF 网络的相应结果。

表 5-5　隐藏层中心大小 $m_1 = 20$，各种正则化参数详细的正确分类概率

	正则化参数，λ					
总体统计	0	0.1	1	10	100	1000
均值	57.49	72.42	74.42	73.80	72.46	72.14
标准偏差	7.47	4.11	3.51	4.17	4.98	5.09
最小	44.20	61.60	65.80	63.10	60.90	60.50
最大	72.70	78.30	78.90	79.20	79.40	79.40

表 5-6　隐层中心大小 $m_1 = 100$，各种正则化参数详细的正确分类概率

	正则化参数，λ					
总体统计	0	0.1	1	10	100	1000
均值	50.58	77.03	77.72	77.87	76.47	75.33
标准偏差	4.70	1.45	0.94	0.91	1.62	2.25
最小	41.00	70.60	75.10	75.10	72.10	70.10
最大	61.30	79.20	79.80	79.40	78.70	78.20

306

图 5-7 显示的是当正则化参数 λ = 10 时由网络输出所形成的决策边界，此时有最优的统计结果。图 5-7 的两部分分别对应于总体中测试表现最好的和最差的网络；图的两部分对应的都是 100 个中心的情况。

比较表 5-5 和 5-6 我们可以发现：

1. 对 $m_1 = 20$ 个和 $m_1 = 100$ 个中心，当 λ = 0 时，网络的分类能力都较差。

2. 正则化方法的使用对 RBF 网络的分类能力有着明显的影响。

3. 当 λ≥0.1 时，网络的分类性能随着 λ 的增加变化不大。在中心为 20 个的情况下，当 λ = 1 时分类性能最佳；在中心为 100 个的情况下，当 λ = 10 时分类性能最佳。

307

4. 当中心数由 20 增加至 100 时，网络的分类性能增加了约 4.5%。

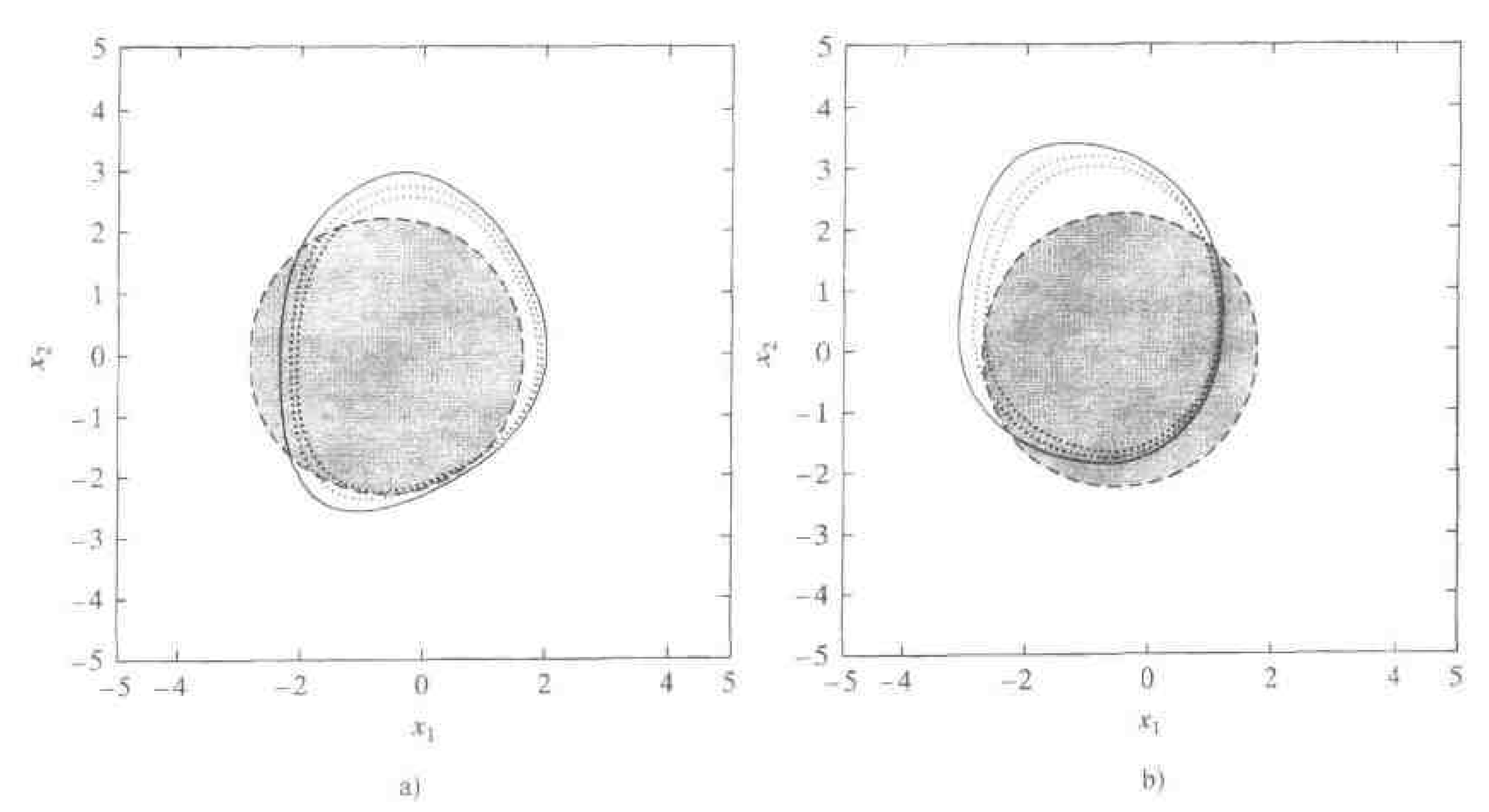


图5-7 基于严格插值法的正则化RBF网络模式分类的试验结果

a)最优解 b)最差解

虚线(实心)圆表示最优Bayes解

5.15 小结和讨论

RBF网络的结构是不常见的，因为隐藏单元的结构与输出单元的结构是完全不同的。由于径向基函数是隐藏单元设计的基础，所以RBF网络的理论与径向基函数理论有着密切的联系，径向基函数理论是数值分析中的一个主要研究领域(Singh，1992)。另外值得注意的是由于输出层的线性权值是可调参数，通过对线性自适应滤波器的有关文献的研究，我们可以得到更多结果。

与采用反向传播算法的多层感知器不同，RBF网络设计采用原理化的方法。特别是5.5节介绍的Tikhonov正则化理论为RBF网络的形成提供坚实的数学基础。在这个理论中Green函数$G(\mathbf{x},\xi)$起着关键作用。作为网络基函数的Green函数形式是由正则化理论应用中的光滑度约束所决定的。由式(5.63)所示的微分算子$\mathbf{D}$指定的光滑度约束将导出多元Gauss函数作为Green函数。微分算子$\mathbf{D}$不同，自然Green函数的形式也不同。记住，当放宽要求基函数比数据点少时，减少计算复杂性就成为确定光滑正则化网络的一个重要因素。这可能是在正则化RBF网络设计中使用其他函数(如习题5.1所描述的薄板样条函数)作为基函数的一个原因。无论选择什么样的函数作为基函数，为了将正则化理论的优点完全应用于RBF网络的设计中，我们都需要一个原理化的方法来估计正则化参数λ。5.9节所介绍的广义交叉确认满足了这个需要。使用广义交叉确认的理论基础是渐近的，这就要求有一个足够大的训练集合，才能得到理想的λ的估计值。

另一个设计RBF网络的原理化方法是通过核回归来实现的。该方法使用密度估计，对于密度估计，径向基函数之和等于1。多元Gauss分布提供满足这一要求的便利方法。

总之，Gauss型RBF网络所实现的输入-输出映射与混合专家系统所实现的输入-输出映射很相似。后一模型将在第7章中介绍。

注释和参考文献

[1] 径向基函数首先是在解决实多变量插值问题时提出的。这方面的早期工作在 Powell (1985) 中有所论述。现在径向基函数是数值分析研究中的一个主要方向。

Broomhead and Lowe(1988)首先将径向基函数应用于神经网络设计。Poggio and Girosi (1990a)在径向基函数网络的理论与设计中也作出了重大贡献。后一篇论文强调将正则化理论应用于这类神经网络，以提高对新数据的泛化能力。 308

[2] Cover 定理的证明遵循如下考虑(Cover,1965)：

- Schlafli *定理或函数计数定理*：对 m_1 维欧几里德空间上的 N 个处于一般位置的向量进行二分，可得到的齐次线性可分的二分方式的数目等于

$$C(N,m_1) = 2\sum_{m=0}^{m_1-1}\binom{N-1}{m}$$

如果每一个含有 m_1 个或小于 m_1 个的向量子集都是线性独立的，就说 m_1 维 Euclid 空间上的集合$\mathscr{X} = \{\mathbf{x}_i\}_{i=1}^{N}$处于一般位置。

- $\mathscr{X}$的联合概率分布的*反射不变性*：一个随机二分是可分的概率(在$\mathscr{X}$的条件下)等于$\mathscr{X}$的一个特定二分(所有的 N 个向量都属于一类)的非条件概率。

函数计数定理由 Camer on (1960)、Joseph(1960)和 Winder(1961)以不同的形式独立证明，并应用于特定的感知器配置(即线性阈值单元)。在 Cover(1968)中这个定理还被用于根据所有可调参数的总数估计感知器网络的能力，它的下界是 $N/(1+\log_2 N)$，其中 N 是输入模式的数量。

[3] 先验知识嵌入输入 - 输出映射的另一种正则化方法是通过使用 Bayes 插值理论；详细了解这方面的资料请参看文献 MacKay(1992a,b)和 Neal(1995)。

[4] 正则化理论的创立主要归功于 Tikhonov(1963)。Phillips(1962)也曾经阐述过相似的理论。因此有时我们也称这一理论为 *Tikhonov-Phillips 正则化*。

在保险统计文献中一种正则化形式曾经在 Whittaker(1923)讨论过，在那里考虑的光滑过程被称为*校准*(graduation)或者*调整*(adjustment)。

以书的形式讨论正则化理论，可以参考 Tikhonov and Arsenin (1977)、Mozorov(1993)及 Kirch(1996)。

[5] 函数空间的概念是 Hilbert 在对一类积分方程所做的基本研究的结果中提出的。当 Fredholm 积分的创始人 Fredholm 用本质为代数的语言提出问题时，Hilbert 意识到这个问题与多维欧几里德空间上的二阶曲面的解析几何理论有着紧密的联系(Lanczos, 1964)。

[6] *赋范空间*是一个定义了实值函数 $\|\mathbf{x}\|$ 的线性向量空间，该实值函数称为 $\mathbf{x}$ 的范数。范数 $\|\mathbf{x}\|$ 具有如下性质：

$$\|\mathbf{x}\| > 0(\mathbf{x}\neq\mathbf{0}),\ \|\mathbf{0}\| = 0$$

$$\|a\mathbf{x}\| = |a|\cdot\|\mathbf{x}\|(a = \text{常数}),\ \|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$$

范数 $\|\mathbf{x}\|$ 的意义是向量 $\mathbf{x}$ 的长度。

[7] 严格说来，我们要求负责产生数据的函数 $f(\mathbf{x})$属于具有 Dirac delta 广义函数 δ 再生核形式的*再生核 Hilbert 空间*(reproducing kernel Hilbert space, RKHS)(Tapia and Thompson, 309

1978)。这样做是因为我们要求 Dirac delta 广义函数 δ 为递减的、无限连续可微的函数，即广义函数 Schwarz 理论的经典测试函数空间$\mathcal{S}$具有有限的 $\mathbf{D}$－诱导范数，表示为

$$H_p = \{f \in \mathcal{S}: \|\mathbf{D}f\| < \infty\}$$

一般说来，当提到 Hilbert 空间时，工程师们总是只想到 L_2 空间，可能因为 L_2 空间与任何 Hilbert 空间同构。但是范数才是 Hilbert 空间最重要的特性，且等距同构(保范意义下的同构)要比简单的加性同构重要得多(Kailath, 1974)。RKHS 理论说明除了 L_2 空间外，还有许多其他不同但是很有用的 Hilbert 空间。关于 RKHS 的指导性综述，参看 Kailath(1971)。

[8] 内积空间是一个线性向量空间，空间中的向量 $\mathbf{u}$ 和 $\mathbf{v}$ 的内积用$(\mathbf{u},\mathbf{v})$表示，满足如下性质：

$$(\mathbf{u},\mathbf{v}) = (\mathbf{v},\mathbf{u}), (a\mathbf{u},\mathbf{v}) = a(\mathbf{u},\mathbf{v}), (a = \text{常数}),$$
$$(\mathbf{u}+\mathbf{v},\mathbf{w}) = (\mathbf{u},\mathbf{w}) + (\mathbf{v},\mathbf{w}), (\mathbf{u},\mathbf{u}) > 0(\mathbf{u} \neq \mathbf{0})$$

如果一个内积空间$\mathcal{H}$中的每一个 Cauchy 序列都按范数收敛于$\mathcal{H}$中的一个点，就说该内积空间是完备的，并且称其为 Hilbert 空间。向量序列$\{\mathbf{x}_n\}$为 Cauchy 序列是指如果对于每一个 $\varepsilon > 0$，都存在一个数 M，使得对所有$(m,n) > M$ 有 $\|\mathbf{x}_m - \mathbf{x}_n\| < \varepsilon$。

[9] 在 Girosi et al.(1995)中，给出得到了式(5.55)的不同方法；该方法直接将正则化项$\mathscr{E}_c(F)$与逼近函数 $F(\mathbf{x})$的光滑性联系起来。

光滑性可看作函数振荡性的度量。特别地，如果某一函数与另一函数相比具有较小的振荡性，我们就说这一函数比另一函数光滑。换句话说，一个函数越光滑，它所含的高频分量就越小。考虑光滑性的这个度量，令 $\tilde{F}(\mathbf{s})$为 $F(\mathbf{x})$的多维 Fourier 变换，$\mathbf{s}$ 表示多维变换变量。令 $H(\mathbf{s})$表示一个正函数，当$\|\mathbf{s}\|$趋向于无穷时这个函数趋向于零，即 $1/H(\mathbf{s})$表示一个"高通滤波器"的作用。那么，根据 Girosi et al.(1995)，我们可以用一个光滑性泛函来表示正则化项：

$$\mathscr{E}_C(F) = \frac{1}{2}\int_{\mathbb{R}^{m_0}} \frac{|\tilde{F}(\mathbf{s})|^2}{H(\mathbf{s})} d\mathbf{s}$$

其中 m_0 是 $\mathbf{x}$ 的维数。根据 Fourier 理论中的 Parseval 定理，这个泛函是高通滤波器 $1/H(\mathbf{s})$的输出功率的一种度量。这样，将正则化问题映射到 Fourier 领域并且利用 Fourier 变换的性质，我们就可以得到式(5.55)所示的解。

[10] 线性微分算子的最一般的形式为

$$\mathbf{D} = p(x_1, x_2, \cdots, x_{m_0}) \frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k}, a + b + \cdots + k = n$$

其中$x_1, x_2, \cdots, x_{m_0}$是向量 $\mathbf{x}$ 的分量，$p(x_1, x_2, \cdots, x_{m_0})$是某个关于这些分量的函数。算子 $\mathbf{D}$ 的伴随算子为(Morse and Feshback, 1953)

310

$$\tilde{\mathbf{D}} = (-1)^n \frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k}[p(x_1, x_2, \cdots, x_{m_0})], a + b + \cdots + k = n$$

[11] 为了从通常的交叉确认得到广义交叉确认，我们先考虑在 Wahba(1990)中的一个岭回归问题(ridge regression problem)：

$$\mathbf{y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon} \tag{1}$$

其中 $\mathbf{X}$ 是一个 $N \times N$ 阶的输入矩阵，噪声向量 $\boldsymbol{\varepsilon}$ 具有零均值，且其协方差矩阵等于 $\sigma^2\mathbf{I}$。对 $\mathbf{X}$ 进行奇异值分解有

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

式中 $\mathbf{U}$ 和 $\mathbf{V}$ 是正交矩阵，$\mathbf{D}$ 是对角阵。令

$$\tilde{\mathbf{y}} = \mathbf{U}^T\mathbf{y}, \qquad \boldsymbol{\beta} = \mathbf{V}^T\boldsymbol{\alpha}, \qquad \tilde{\boldsymbol{\varepsilon}} = \mathbf{U}^T\boldsymbol{\varepsilon}$$

我们可以用 $\mathbf{U}$ 和 $\mathbf{V}$ 将式(1)转变为

$$\tilde{\mathbf{y}} = \mathbf{D}\boldsymbol{\beta} + \tilde{\boldsymbol{\varepsilon}} \tag{2}$$

选择对角矩阵 $\mathbf{D}$(注意不要与微分算子混淆)使其奇异值成对出现。这样就有一个正交矩阵 $\mathbf{W}$，使 $\mathbf{W}\mathbf{D}\mathbf{W}^T$ 是轮换矩阵；即

$$\mathbf{A} = \mathbf{W}\mathbf{D}\mathbf{W}^T = \begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ a_{N-2} & a_{N-1} & \cdots & a_{N-3} \\ \vdots & \vdots & & \vdots \\ a_1 & a_2 & \cdots & a_0 \end{bmatrix}$$

它的对角线元素为常数。令

$$\mathbf{z} = \mathbf{W}\tilde{\mathbf{y}}, \qquad \boldsymbol{\gamma} = \mathbf{W}\boldsymbol{\beta}, \qquad \boldsymbol{\xi} = \mathbf{W}\tilde{\boldsymbol{\varepsilon}}$$

则式(2)变换为

$$\mathbf{z} = \mathbf{A}\boldsymbol{\gamma} + \boldsymbol{\xi} \tag{3}$$

对角矩阵 $\mathbf{D}$ 具有矩阵“最大解耦”(maximally uncoupled)行，而轮换矩阵 $\mathbf{A}$ 具有“最大耦合”(maximally coupled)行。

按照上述变换，我们可以陈述广义交叉确认等价于将式(1)所示的岭回归问题变换为式(3)所示的最大耦合形式，然后对 $\mathbf{z}$ 进行一般的交叉确认，最后将其变换为原坐标系统(Wahba，1990)。 311

[12] 基于 1990 年提出的报告，在 Powell(1992)贡献部分的附录中，对 A.C.Brown 得到的结果给予了高度评价。很明显在 1981 年得出的这个结果说明一个 RBF 网络可以实现任意一个函数从 $\mathbb{R}^{m_0}$ 的闭域上到 $\mathbb{R}$ 的映射。

Hartman et al.(1990)讨论 Gauss 函数以及在凸的 $\mathbb{R}^{m_0}$ 紧子集上的逼近，在那里证明具有单隐藏层且激活函数为 Gauss 函数的 RBF 网络是一个通用逼近器。然而，对 RBF 网络的通用逼近性质的最严格的证明在 Park and Sandberg(1991)中；这后一个工作在 Hartman et al. 的论文发表前就已经完成了。

[13] 让 Ω 为 $\mathbb{R}^n$ 上的一个有界区域，其边界为 Γ。考虑 $\mathcal{G}$ 表示在 $\bar{\Omega} = \Omega + \Gamma$ 上具有连续梯度的连续实值函数的集合。双线性形式

$$\int_{\Omega} (\mathrm{grad}\,u : \mathrm{grad}\,v + uv)\, d\mathbf{x}$$

一定是 $\mathcal{G}$ 上的一个内积。由这个内积产生的范数完备的空间 $\mathcal{G}$ 称为 Sobolev 空间(Debnath and Mikusiński，1990)。Sobolev 空间在偏微分方程理论上有着重要的作用，因此是 Hilbert 空间的一个重要的例子。

[14] 关于 Parzen-Rosenblatt 的密度估计器的渐近无偏性的证明，参看 Parzen(1962)和 Cacoullos (1966)。

[15] Nadaraya-Watson 回归估计器在统计学文献中已是一个广泛研究的主题。从更广的意义上说，非参数泛函估计在统计学中占有中心地位；参看 Härdle(1990)及 Roussas(1991)的论文集。

习题

径向基函数

5.1 一个薄板样条函数可以写成

$$\varphi(r) = \left(\frac{r}{\sigma}\right)^2 \log\left(\frac{r}{\sigma}\right) \qquad \text{对于某个 } \sigma > 0 \text{ 及 } r \in \mathbb{R}$$

证明可以用此函数作为具有平移及旋转不变性的 Green 函数。

5.2 在5.8节中给出的对图5-6所示的 RBF 网络的权值向量 $\mathbf{w}$ 的值集合，对 XOR 问题提出一组可能的解。试求另一组能解决该问题的权值向量 $\mathbf{w}$ 的值。

5.3 在5.8节中我们给出了用具有两个隐藏单元的 RBF 网络解决 XOR 问题的解。在这个习题中，我们考虑用四个隐藏单元精确求解该问题，每个径向基函数的中心由每一个输入数据点决定。四个可能的输入模式为(0,0)、(0,1)、(1,1)和(1,0)，它们表示一个正方形环形排序的四个角。

312

(a)求上述 RBF 网络的插值矩阵 $\boldsymbol{\Phi}$ 及其逆 $\boldsymbol{\Phi}^{-1}$。

(b)计算该网络的输出层的线性权值。

5.4 Gauss 函数是仅有的可因式分解的径向基函数。利用 Gauss 函数的这个性质证明定义为多元 Gauss 分布的 Green 函数可分解成

$$G(\mathbf{x},\mathbf{t}) = \prod_{i=1}^{m} G(x_i, t_i)$$

其中 x_i 和 t_i 是 $m \times 1$ 维向量 $\mathbf{x}$ 和 $\mathbf{t}$ 的第 i 个分量。

正则化网络

5.5 考虑代价泛函

$$\mathscr{E}(F^*) = \sum_{i=1}^{N}\left[d_i - \sum_{j=1}^{m_1} w_j G(\|\mathbf{x}_j - \mathbf{t}_i\|)\right]^2 + \lambda \|\mathbf{D}F^*\|^2$$

它用到逼近函数

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\|\mathbf{x} - \mathbf{t}_i\|)$$

利用 Frèchet 微分，证明当

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_0)\mathbf{w} = \mathbf{G}^T\mathbf{d}$$

时，代价泛函$\mathscr{E}(F^*)$最小，其中 $N \times m_1$ 维矩阵 $\mathbf{G}$，$m_1 \times m_1$ 维矩阵 $\mathbf{G}_0$，$m_1 \times 1$ 向量 $\mathbf{w}$ 以及 $N \times 1$ 向量 $\mathbf{d}$，分别由式(5.72)、(5.75)、(5.73)及(5.46)定义。

5.6 假设我们定义

$$(\tilde{\mathbf{D}}\mathbf{D})_{\mathrm{U}} = \sum_{k=0}^{\infty}(-1)^k \frac{\nabla_{\mathrm{U}}^{2k}}{k!2^k}$$

其中

$$\nabla_{\mathrm{U}}^2 = \sum_{j=1}^{m_0}\sum_{i=1}^{m_0} u_{ji} \frac{\partial^2}{\partial x_j \partial x_i}$$

$m_0 \times m_0$ 阶矩阵 $\mathbf{U}$ 是一个对称正定矩阵，第 ji 个元素用 u_{ji} 表示。因此存在逆矩阵 $\mathbf{U}^{-1}$，从而可以通过相似变换将其分解成如下形式

$$\mathbf{U}^{-1} = \mathbf{V}^T\mathbf{\Sigma}\mathbf{V} = \mathbf{V}^T\mathbf{\Sigma}^{1/2}\mathbf{\Sigma}^{1/2}\mathbf{V} = \mathbf{C}^T\mathbf{C}$$

313

式中 $\mathbf{V}$ 是一个正交矩阵，$\mathbf{\Sigma}$ 是对角矩阵，$\mathbf{\Sigma}^{1/2}$ 是 $\mathbf{\Sigma}$ 的平方根，矩阵 $\mathbf{C}$ 定义

$$\mathbf{C} = \mathbf{\Sigma}^{1/2}\mathbf{V}$$

问题相当于求 Green 函数 $G(\mathbf{x},\mathbf{t})$ 满足下列条件（在广义函数的意义下）：

$$(\tilde{\mathbf{D}}\mathbf{D})_{\mathrm{U}}G(\mathbf{x},\mathbf{t}) = \delta(\mathbf{x}-\mathbf{t})$$

用多维 Fourier 变换解关于 $G(\mathbf{x},\mathbf{t})$ 的方程，证明其解为

$$G(\mathbf{x},\mathbf{t}) = \exp\left(-\frac{1}{2}\|\mathbf{x}-\mathbf{t}\|_C^2\right)$$

其中

$$\|\mathbf{x}\|_C^2 = \mathbf{x}^T\mathbf{C}^T\mathbf{C}\mathbf{x}$$

5.7 考虑一个定义如下的正则化项：

$$\int_{\mathbb{R}^{m_0}}\|\mathbf{D}F(\mathbf{x})\|^2 d\mathbf{x} = \sum_{k=0}^{\infty} a_k \int_{\mathbb{R}^{m_0}}\|\mathbf{D}^k F(\mathbf{x})\|^2 d\mathbf{x}$$

其中

$$a_k = \frac{\sigma^{2k}}{k!\,2^k}$$

线性微分算子 D 由梯度算子 ∇ 和拉普拉斯算子 ∇^2 定义如下：

$$D^{2k} = (\nabla^2)^k$$

且

$$D^{2k+1} = \nabla(\nabla^2)^k$$

证明

$$\mathbf{D}F(\mathbf{x}) = \sum_{k=0}^{\infty}\frac{\sigma^{2k}}{k!\,2^k}\nabla^{2k}F(\mathbf{x})$$

5.8 在第 5.5 节中，我们由式(5.65)的关系导出了关于 $F_\lambda(\mathbf{x})$ 的式(5.66)。在这个问题中我们希望从由式(5.65)开始利用多维 Fourier 变换导出式(5.66)。利用 Green 函数 $G(\mathbf{x})$ 的多维 Fourier 变换的定义

$$G(\mathbf{s}) = \int_{\mathbb{R}^{m_0}} G(\mathbf{x})\exp(-i\mathbf{s}^T\mathbf{x})d\mathbf{x}$$

完成推导，其中 $i=\sqrt{-1}$，$\mathbf{s}$ 是 m_0 维的变换变量。

5.9 考虑式(5.95)所描述的非线性回归问题。令 a_{ik} 表示矩阵 $(\mathbf{G}+\lambda\mathbf{I})^{-1}$ 的第 ik 个元素。那么，由式(5.58)出发，证明回归函数 $f(\mathbf{x})$ 的估计可以表示为

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{N}\Psi(\mathbf{x},\mathbf{x}_k)y_k$$

其中 y_k 是对应于模型输入 $\mathbf{x}_k$ 的输出，且

$$\Psi(\mathbf{x},\mathbf{x}_k) = \sum_{i=1}^{N} G(\|\mathbf{x}-\mathbf{x}_i\|)a_{ik}, \qquad k = 1,2,\cdots,N$$

314

上式中 $G(\|\cdot\|)$ 是 Green 函数。

5.10 样条函数是分段多项式逼近器的例子（Schumaker, 1981）。样条方法的基本思想如下：将一个被逼近区域用节点分为有限个子区域；节点可以是固定的，这样逼近器就是线性参数化的；节点也可以是可变的，这样逼近器就是非线性参数化的。在这两种情况下，在每一个逼近区域中使用一个阶数最高为 n 的多项式，且要求整个函数必须是 $n-1$ 次可微的。

多项式样条函数是相对光滑函数，容易在计算机上存储、操作及计算。

在实际使用的样条函数中，三次样条函数可能是应用最广泛的。一个一维输入的三次样条函数的代价泛函定义如下：

$$\mathscr{E}(f) = \frac{1}{2}\sum_{i=1}^{N}[y_i - f(x_i)]^2 + \frac{\lambda}{2}\int_{x_1}^{x_N}\left[\frac{d^2 f(x)}{dx^2}\right]^2 dx$$

其中 λ 在样条函数中表示光滑性参数。

(a)验证这个问题解 $f_\lambda(x)$ 的如下性质：

(1)两个相续的 x 节点值之间 $f_\lambda(x)$ 是一个三次多项式。

(2)$f_\lambda(x)$ 及前两阶导数都是连续的，除其二阶导数值在边界点为零外。

(b)因为 $\mathscr{E}(f)$ 有惟一最小值，所以我们必须有

$$\mathscr{E}(f_\lambda + \alpha g) \geqslant \mathscr{E}(f_\lambda)$$

其中 g 是与 f_λ 一类的二次可微函数，α 为任意实值常数。这意味着 $\mathscr{E}(f_\lambda + \alpha g)$ 作为 α 的函数在 $\alpha = 0$ 局部最小。因此，证明

$$\int_{x_1}^{x_N}\left(\frac{d^2 f_\lambda(x)}{dx^2}\right)\left(\frac{d^2 g(x)}{dx^2}\right)dx = \frac{1}{2}\sum_{i=1}^{N}[y - f_\lambda(x_i)]g(x_i)$$

上式是关于三次样条问题的 Euler-Lagrange 方程。

逼近速度

5.11 设计 Guass 型 RBF 网络逼近属于某一 Sobolev 空间的一个回归函数时，式(5.124)定义泛化误差的上界。利用这个上界推导式(5.125)的公式，该式表示这个网络对应于某一特定大小的训练样本的最佳网络大小。

核估计

5.12 假设给你一个“无噪声”训练集合 $\{f(\mathbf{x}_i)\}_{i=1}^{N}$，要求设计一个神经网络，能推广到由于受加噪声的干扰而不属于训练集合的那些样本点上。令 $F(\mathbf{x})$ 表示该网络所实现的逼近函数，它使期望平方误差

315

$$J(F) = \frac{1}{2}\sum_{i=1}^{N}\int_{\mathbb{R}^{m_0}}[f(\mathbf{x}_i) - F(\mathbf{x}_i, \boldsymbol{\xi})]^2 f_{\xi}(\boldsymbol{\xi})\,d\boldsymbol{\xi}$$

成为最小，其中 $f_\xi(\boldsymbol{\xi})$ 是输入空间 $\mathbb{R}^{m_0}$ 上的一个噪声分布的概率密度函数。证明这个最小平方问题的解为(Webb, 1994)

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{N} f(\mathbf{x}_i) f_\xi(\mathbf{x} - \mathbf{x}_i)}{\sum_{i=1}^{N} f_\xi(\mathbf{x} - \mathbf{x}_i)}$$

比较这个估计器和 Nadaraya-Watson 回归估计器。

中心的监督选择

5.13 考虑代价泛函

$$\mathscr{E} = \frac{1}{2}\sum_{j=1}^{N} e_j^2$$

其中

$$e_j = d_j - F^*(x_j) = d_j - \sum_{i=1}^{m_1} w_i G(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i})$$

式中的自由参数为线性权值 w_i，Green 函数的中心 $\mathbf{t}_i$ 以及协方差矩阵的逆 $\mathbf{\Sigma}_i^{-1}=\mathbf{C}_i^T\mathbf{C}_i$，其中 $\mathbf{C}_i$ 是范数加权矩阵。要求找到使代价泛函$\mathscr{E}$最小的自由参数。推导下列偏导数

(a) $\dfrac{\partial \mathscr{E}}{\partial w_i} = \sum_{j=1}^{N} e_j G(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i})$

(b) $\dfrac{\partial \mathscr{E}}{\partial \mathbf{t}_i} = 2w_i \sum_{j=1}^{N} e_j G'(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i})\mathbf{\Sigma}_i^{-1}(\mathbf{x}_j - \mathbf{t}_i)$

(c) $\dfrac{\partial \mathscr{E}}{\partial \mathbf{\Sigma}_i^{-1}} = -w_i \sum_{j=1}^{N} e_j G'(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i})\mathbf{Q}_{ji}$

其中 $G'(\cdot)$是 $G(\cdot)$对其自变量的导数，且

$$\mathbf{Q}_{ji} = (\mathbf{x}_j - \mathbf{t}_i)(\mathbf{x}_j - \mathbf{t}_i)^T$$

关于一个标量对一个向量的求导数规则，参看第 3 章的注释[2]。

计算机实验

5.14 在本题中，我们将继续 5.13 节中的计算机实验，在设计作为二值模式分类器的 RBF 网络时讨论随机选取中心的情况。实验的目的是为了证明以这种方式训练的网络的泛化能力相当好。

设计的网络是为了解决 5.13 节中的二值模式分类问题，要求分类的数据是从一个具有两个等概率的部分重叠二维 Gauss 分布的混合模型中抽取的。其中一个 Gauss 分布的均值向量为$[0,0]^T$，公共方差为 1；另一个 Gauss 分布的均值向量为$[0,2]^T$，公共方差为 4。该分类的决策规则为“选择具有最大函数输出的类”。 316

(a)随机选取 $m_1=20$ 个中心，在正则化参数 λ 分别为 0，0.1，1，10，100 和 1000 的情况下计算均值、标准偏差以及正确分类概率 P_c 的最小值和最大值。为了计算总体统计量，对每一个总体利用 50 个独立的网络分别测试，每次都是对一个固定的具有 1000 个模式的参考集合进行测试。

(b)构造按(a)所述配置计算的当正则化参数 $\lambda=1$ 时的决策边界。

(c)当中心数 $m_1=10$ 时(随机选择中心)，重复(a)的计算。

(d)根据结果，讨论将随机选择中心作为 RBF 网络设计方法的优点，以及当网络作为模式分类器时正则化在性能方面所起的作用。

(e)将所得结果与 5.13 节中用严格插值法所得的结果进行比较。特别地，确定随机选择中心的方法对正则化参数更不敏感。

5.15 也许可以说，在 5.13 节对一对 Gauss 分布类进行分类的计算机实验中，由于用 Gauss 径向基函数逼近固有的 Gauss 类条件分布，所以 RBF 网络有较好的性能。在本题中我们将用计算机试验研究设计一个严格插值的 Gauss 型 RBF 网络，Gauss 分布为明显不连续的类条件分布。特别地，考虑两个等可能的类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 的分布：

- $U(\mathscr{C}_1)$，其中$\mathscr{C}_1 \triangleq \Omega_1$ 是一个半径为 $r=2.34$、中心在 $\mathbf{x}_c=[-2,30]^T$ 的圆
- $U(\mathscr{C}_2)$，其中$\mathscr{C}_2 \subset \mathbb{R}^2$是一个中心在 $\mathbf{x}_c$、边长为 $r=\sqrt{2\pi}$的正方形区域

这里 $U(\Omega)$表示一个在 $\Omega \subset \mathbb{R}^2$上的均匀分布。这些参数的选取使得类$\mathscr{C}_1$ 的决策区域与 5.13 节中用 Gauss 分布情况时的决策区域相同。研究使用正则化作为一种手段，提高利用严格插值的 Gauss 型 RBF 网络的分类性能。 317



第6章 支持向量机

6.1 简介

在第4章，我们研究了由反向传播算法训练的多层感知器。在第5章，我们研究了另一类分层前馈网络，即径向基函数网络。这两种神经网络按它们自己的方式都是通用逼近器。在这一章，我们将讨论另一种通用的前馈网络的类型，称为*支持向量机*(support vector machine,SVM)，由 Vapnik 首先提出(Boser, Guyon, and Vapnik, 1992; Cortes and Vapnik, 1995; Vapnik, 1995, 1998)。像多层感知器网络和径向基函数网络，支持向量机能用于模式分类和非线性回归。

当然，支持向量机是一种*线性机器*，有一些很好的特性。为了解释它怎样工作，从模式分类中出现的可分模式的情况开始可能是最容易的。在此背景下，支持向量机的主要思想是建立一个超平面作为决策曲面，使得正例和反例之间的隔离边缘被最大化。通过使用在第2章中讨论过的基于统计学习理论的原理性方法，机器获得了这个想要的特性。更精确地说，支持向量机是*结构风险最小化方法*的近似实现。这个归纳原理是基于这样的事实，学习机器在测试数据上的误差率(即泛化误差率)以训练误差率和一个依赖于 VC 维数(Vapnik-Chervonenkis dimension)的项的和为界；在可分模式情况下，支持向量机对于前一项的值为零，并且使第二项最小化。因此，尽管它不利用问题的领域知识，在模式分类问题上支持向量机能提供好的泛化性能。这个属性是支持向量机特有的。

在"支持向量"$\mathbf{x}_i$ 和输入空间抽取的向量 $\mathbf{x}$ 之间的内积核这一个概念是构造支持向量机学习算法的关键。支持向量是由算法从训练数据中抽取的小的子集构成。依赖于这个内积核的不同产生方式，可能建立不同的学习机器，由它们自己的非线性决策曲面所表征。尤其是，可以使用支持向量学习算法来构建学习机器中的下面三种类型： 318

- 多项式学习机器
- 径向基函数网络
- 两层感知器(即有单独隐藏层)

也就是说，对于这些前馈网络中的每一个，利用给定的训练数据集我们可以使用支持向量学习算法来实现学习过程，自动决定要求隐藏单元的数目。用另一种方式陈述：由于反向传播算法专门为训练多层感知器设计，所以支持向量学习算法是一种更一般化的算法，因为它有更广泛的应用。

本章的组织

本章的主体组织为三部分。在第一部分中，我们描述支持向量机背后的基本思想。特别地，在6.2节讨论对于简单的线性可分模式情况下最优超平面的构建。接着在6.3节考虑更复杂的不可分模式的情况。

按照这样做，我们为本章的第二部分铺平道路，这部分给出支持向量机解决模式识别任

务的详细讨论。这个工作在6.4节完成。在第6.5节再回到XOR问题，说明支持向量机的构造。在第6.6节再次谈及在第4，5章中研究过的模式分类的计算机实验，从而提供支持向量机与由反向传播算法训练的多层感知器及标准的径向基函数网络之间的一个比较。

本章的最后一部分处理非线性回归问题。在6.7节描述一个损失函数，它非常适合这个问题。然后在第6.8节讨论用于非线性回归的支持向量机的构造。

在第6.9节以一些最终评述结束本章。

6.2 线性可分模式的最优超平面

考虑训练样本$\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$，其中$\mathbf{x}_i$是输入模式的第$i$个例子，$d_i$是对应的期望响应(目标输出)。开始我们假定由子集$d_i=+1$代表的模式(类)和$d_i=-1$代表的模式是"线性可分的"。用于分离的超平面形式的决策曲面方程是

319

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{6.1}$$

其中$\mathbf{x}$是输入向量，$\mathbf{w}$是可调的权值向量，b是偏置。这样我们可以写成

$$\mathbf{w}^T\mathbf{x}_i + b \geq 0, \quad 对于\ d_i=+1 \qquad \mathbf{w}^T\mathbf{x}_i + b < 0, \quad 对于\ d_i=-1 \tag{6.2}$$

在这里作了模式线性可分的假定，以便在相当简单的环境里解释支持向量机背后的基本思想；在第6.3节将放宽这个假定。

对于一个给定的权值向量$\mathbf{w}$和偏置b，由方程(6.1)定义的超平面和最近的数据点之间的间隔被称为分离边缘，用ρ表示。支持向量机的目标是找到一个特殊的超平面，对于这个超平面分离边缘ρ最大。在这个条件下，决策曲面称为最优超平面(optimal hyperplane)。图6-1给出的是二维输入空间中最优超平面的几何结构。

图6-1 线性可分模式最优超平面的思想示意图

设$\mathbf{w}_o$和b_o分别表示权值向量和偏置的最优值。相应地，在输入空间里表示多维线性决策面的最优超平面由

$$\mathbf{w}_o^T\mathbf{x} + b_o = 0 \tag{6.3}$$

定义，它是方程(6.1)的改写。判别函数

320

$$g(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o \tag{6.4}$$

给出从$\mathbf{x}$到最优超平面的距离的一种代数度量(Duda and Hart，1973)。看出这一点的最简单方法或许是将$\mathbf{x}$表达为

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}_o}{\|\mathbf{w}_o\|}$$

其中，$\mathbf{x}_p$是$\mathbf{x}$在最优超平面上的常规投影，r是期望的代数距离；如果$\mathbf{x}$在最优超平面的正面，r是正值；相反如果$\mathbf{x}$在最优超平面的负面，r是负值。因为由定义知$g(\mathbf{x}_p)=0$，由此推出

$$g(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o = r\|\mathbf{w}_o\|$$

或者

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}_o\|} \tag{6.5}$$

尤其，从原点(即 $\mathbf{x}=\mathbf{0}$)到最优超平面的距离由 $b_o/\|\mathbf{w}_o\|$ 给定。如果 $b_o>0$，原点在最优超平面的正面；如果 $b_o<0$，原点在负面；如果 $b_o=0$，最优超平面通过原点。这些代数结果的几何解释在图 6-2 中给出。

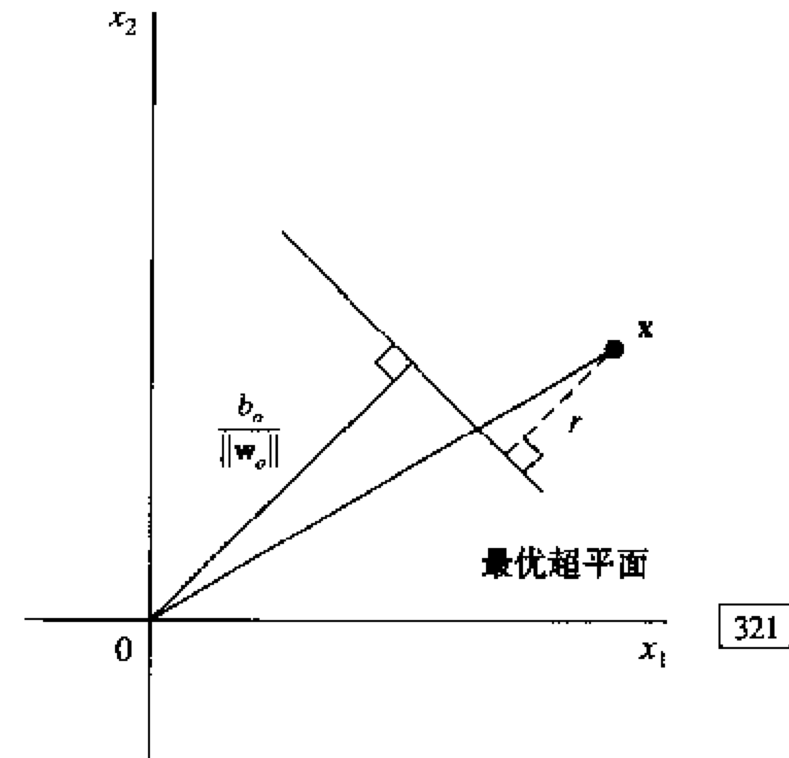


图 6-2 二维情况下点到最优超平面的代数距离的几何解释

现在的问题是对于给定的数据集$\mathcal{T}=\{(\mathbf{x}_i,d_i)\}$，找到最优超平面的参数 $\mathbf{w}_o$ 和 b_o。根据图 6-2 描绘的结果。可以看出一对$(\mathbf{w}_o,b_o)$一定满足条件：

$$\begin{aligned} \mathbf{w}_o^T\mathbf{x}_i + b_o \geqslant 1, \quad & 对于\ d_i=+1 \\ \mathbf{w}_o^T\mathbf{x}_i + b_o \leqslant -1, \quad & 对于\ d_i=-1 \end{aligned} \tag{6.6}$$

注意如果式(6.2)成立，即模式是线性可分的，总可以重新调整 $\mathbf{w}_o$ 和 b_o 的值使得式(6.6)成立；这种重新调整并不改变式(6.3)。

满足式(6.6)第一行或第二行等号情况的特殊数据点$(\mathbf{x}_i,d_i)$称为*支持向量*，“支持向量机”因此得名。这些向量在这类学习机器的运行中起着主导作用。用概念性的术语，支持向量是那些最靠近决策面的数据点，这样这些数据点是最难分类的。因此，它们和决策面的最优位置直接相关。

考虑一个支持向量 $\mathbf{x}^{(s)}$对应于 $d^{(s)}=+1$。然后根据定义，我们有

$$g(\mathbf{x}^{(s)}) = \mathbf{w}_o^T\mathbf{x}^{(s)} \mp b_o = \mp 1 \quad 对于\ d^{(s)}=\mp 1 \tag{6.7}$$

从式(6.5)知从支持向量 $\mathbf{x}^{(s)}$到最优超平面的*代数距离*是

$$r = \frac{g(\mathbf{x}^{(s)})}{\|\mathbf{w}_o\|} = \begin{cases} \dfrac{1}{\|\mathbf{w}_o\|} & 若\ d^{(s)}=+1 \\ -\dfrac{1}{\|\mathbf{w}_o\|} & 若\ d^{(s)}=-1 \end{cases} \tag{6.8}$$

其中加号表示 $\mathbf{x}^{(s)}$在最优超平面的正面，而减号表示 $\mathbf{x}^{(s)}$在最优超平面的负面。让 ρ 表示在两个类之间的*分离边缘*的最优值，其中这两个类构成训练集合$\mathcal{T}$。因此从式(6.8)得到

$$\rho = 2r = \frac{2}{\|\mathbf{w}_o\|} \tag{6.9}$$

式(6.9)说明，最大化两个类之间的分离边缘等价于最小化权值向量 $\mathbf{w}$ 的欧几里德范数。

总之，由式(6.3)定义的最优超平面是惟一的，意味着最优权值向量 $\mathbf{w}_o$ 提供正反例之间的最大可能的分离。这个优化条件是通过最小化权值向量 $\mathbf{w}$ 的欧几里德范数获得的。

用于寻找最优超平面的二次最优化

我们的目标是发展一个计算上有效的过程，通过使用训练样本$\mathcal{T}=\{(\mathbf{x}_i,d_i)\}_{i=1}^N$找到最优超平面，并且满足约束条件

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 \qquad 对于\ i=1,2,\cdots,N \tag{6.10}$$

这个约束条件把式(6.6)两行组合在一起，其中 $\mathbf{w}_o$ 被 $\mathbf{w}$ 来代替。

我们必须解决的约束最优问题现在可陈述如下：

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$，找到权值向量 $\mathbf{w}$ 和偏置 b 的最优值使得它们满足下面的约束条件

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \text{对 } i = 1,2,\cdots,N$$

并且权值向量 $\mathbf{w}$ 最小化代价函数

322

$$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

这里包含比列因子 1/2 是为了表示方便。这个约束优化问题称为原问题(primal problem)。它的特点如下：

- 代价函数 $\Phi(\mathbf{w})$ 是 $\mathbf{w}$ 的凸函数[1]。
- 约束条件关于 $\mathbf{w}$ 是线性的。

因此，我们可以使用 Lagrange 乘子方法解决约束最优问题(Bertsekas, 1995)。

首先，我们建立 Lagrange 函数

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] \tag{6.11}$$

其中辅助非负变量 α_i 称作 Lagrange 乘子。约束最优问题的解由 Lagrange 函数 $J(\mathbf{w}, b, \alpha)$ 的鞍点决定，此函数对 $\mathbf{w}$ 和 b 必定最小化，对 α 必定最大化。$J(\mathbf{w}, b, \alpha)$ 对 $\mathbf{w}$ 和 b 求微分并置结果等于零，我们得到下面两个最优化条件：

条件 1：$\dfrac{\partial J(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = \mathbf{0}$

条件 2：$\dfrac{\partial J(\mathbf{w}, b, \alpha)}{\partial b} = 0$

应用最优化条件 1 到式(6.11)的 Lagrange 函数，得到(在重新安排项之后)

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i d_i \mathbf{x}_i \tag{6.12}$$

应用最优条件 2 到式(6.11)的 Lagrange 函数，得到

$$\sum_{i=1}^{N}\alpha_i d_i = 0 \tag{6.13}$$

解向量 $\mathbf{w}$ 定义为 N 个训练样本的展开。但是注意，尽管由于 Lagrange 函数的凸性这个解是惟一的，但并不能认为 Lagrange 系数 α_i 亦是惟一的。

在这里同样重要的是注意，在鞍点对每一个 Lagrange 乘子 α_i，乘子与它相应的约束的乘积为零，表示为

323

$$\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] = 0 \qquad \text{对于 } i = 1,2,\cdots,N \tag{6.14}$$

因此，只有这些精确满足式(6.14)的乘子才能假定非零值。这个性质是从最优化理论的 Kuhn-Tucker 条件得出的(Fletcher, 1987; Bertsekas, 1995)。

就像早先提到的，原问题是处理凸代价函数和线性约束。给定这样一个约束最优化问题，可能构造另一个问题，称为对偶问题(dual problem)。这第二个问题与原问题有同样的最优值，但由 Lagrange 乘子提供最优解。特别地，可以陈述对偶定理如下(Bertsekas, 1995)：

(a)如果原问题有最优解，对偶问题也有最优解，并且相应的最优值是相同的。

(b)为了使得 $\mathbf{w}_o$ 为原问题的一个最优解和 α_o 为对偶问题的一个最优解的充分必要条件是 $\mathbf{w}_o$ 对原问题是可行的，并且

$$\Phi(\mathbf{w}_o) = J(\mathbf{w}_o, b_o, \alpha_o) = \min_{\mathbf{w}} J(\mathbf{w}, b_o, \alpha_o)$$

为了说明对偶问题是我们原问题的前提，我们首先逐项展开式(6.11)如下：

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i d_i \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i d_i + \sum_{i=1}^{N}\alpha_i \tag{6.15}$$

按照式(6.13)最优条件的性质，式(6.15)右端第三项为零。而且从式(6.12)我们有

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N}\alpha_i d_i \mathbf{w}^T\mathbf{x}_i = \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

因此，目标函数设置为 $J(\mathbf{w}, b, \alpha) = Q(\alpha)$，可以改写式(6.15)为

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j \tag{6.16}$$

其中 α_i 是非负的。

现在可以陈述对偶问题：

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$，寻找最大化目标函数

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

的 Lagrange 乘子 $\{\alpha_i\}_{i=1}^{N}$，满足约束条件

(1) $\sum_{i=1}^{N}\alpha_i d_i = 0$

(2) $\alpha_i \geqslant 0$　　对于 $i = 1, 2, \cdots, N$

注意，对偶问题完全是根据训练数据来表达的。而且，函数 $Q(\alpha)$ 的最大化仅依赖于输入模式点积的集合 $\{\mathbf{x}_i^T\mathbf{x}_j\}_{(i,j)=1}^{N}$。

确定用 $\alpha_{o,i}$ 表示的最优 Lagrange 乘子后，可以用式(6.12)计算最优权值向量 $\mathbf{w}_o$，并写成

$$\mathbf{w}_o = \sum_{i=1}^{N}\alpha_{o,i} d_i \mathbf{x}_i \tag{6.17}$$

为了计算最优偏置 b_o，可以使用获得的 $\mathbf{w}_o$，并对于一个正的支持向量利用式(6.7)，这样有

$$b_o = 1 - \mathbf{w}_o^T\mathbf{x}^{(s)} \qquad 对于\ d^{(s)} = 1 \tag{6.18}$$

324

最优超平面的统计特性

从第 2 章给出的统计学习理论，回忆学习机器的 VC 维决定逼近函数的嵌套结构应该使用的方式。我们也知道在 m 维空间分离超平面集的 VC 维为 $m+1$。然而，为了应用第 2 章描述的结构风险最小化的方法，我们需要建立 VC 维数变化的分离超平面集合，使得经验风险(即训练分类误差)和 VC 维数同时最小化。在支持向量机里，通过约束权值向量 $\mathbf{w}$ 的 Euclid 范数对分离超平面集合施加一个结构。特别地，我们可以陈述如下定理(Vapnik, 1995,

1998)：

令 D 表示包括所有输入向量$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ 的最小球的直径。由方程

$$\mathbf{w}_o^T\mathbf{x} + b_o = 0$$

描述的最优超平面集合，有一个 VC 维数 h 的上界为

$$h \leqslant \min\left\{ \left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0 \right\} + 1 \tag{6.19}$$

其中顶符号$\lceil \cdot \rceil$表示大于等于所包含的数值的最小整数，ρ 是等于 $2/\|\mathbf{w}_o\|$ 的分离边缘，m_0 是输入空间的维数。

这个定理告诉我们，可以试验控制最优超平面的 VC 维数(即复杂性)，通过正确选择分离边缘 ρ，它与输入空间的维数 m_0 无关。

于是假定，我们有一个通过分离超平面描述的嵌套结构如下：

$$S_k = \{\mathbf{w}^T\mathbf{x} + b : \|\mathbf{w}\|^2 \leqslant c_k\}, \qquad k = 1,2,\cdots \tag{6.20}$$

由 VC 维数 h 在式(6.19)定义的上界，在式(6.20)中描述的嵌套结构可以通过分离边缘改写为等价形式

$$S_k = \left\{ \left\lceil \frac{r^2}{\rho^2} \right\rceil + 1 : \rho^2 \geqslant a_k \right\}, \qquad k = 1,2,\cdots \tag{6.21}$$

其中 a_k 和 c_k 都是常数。

从第 2 章我们也知道，为了得到较好的泛化能力应该选择一个特殊的结构，根据结构风险最小化原则，它应有最小的 VC 维数和训练误差。从式(6.19)和(6.21)中我们发现通过使用最优超平面(即利用具有最大分离边缘 ρ 的分离超平面)，这个要求可以被满足。等价地，根据式(6.9)应该使用具有最小欧几里德范数的最优权值向量 $\mathbf{w}_o$。因此，最优超平面作为线性可分模式决策面的选择，不仅直观上满足而且完全符合支持向量机的结构风险最小化的原理。

325

6.3 不可分模式的最优超平面

到目前为止讨论集中在线性可分模式的情况。在这一节我们考虑更难的不可分模式的情况。给定这样一组训练数据，不可能建立一个不具有分类误差的分离超平面。然而，我们希望找到一个最优超平面，它对整个训练集合平均的分类误差的概率达到最小。

在类之间的分离边缘称为是*软的*，如果数据点$(\mathbf{x}_i, d_i)$不满足下面的条件(见式(6.10))：

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant +1, \qquad i = 1,2,\cdots,N$$

这种违反条件以下面两种方式之一出现：

- 数据点$(\mathbf{x}_i, d_i)$落在分离区域之内，但在决策面正确的一侧，如图 6-3a 所示。
- 数据点$(\mathbf{x}_i, d_i)$落在决策面错误的一侧，如图 6-3b 所示。

注意，在情况 1 我们有正确的分类，但在情况 2 分类是错误的。

为了建立不可分离数据点正式处理的阶段，我们引入一组新的非负标量变量$\{\xi_i\}_{i=1}^N$到分离超平面(即决策面)的定义中，表示为

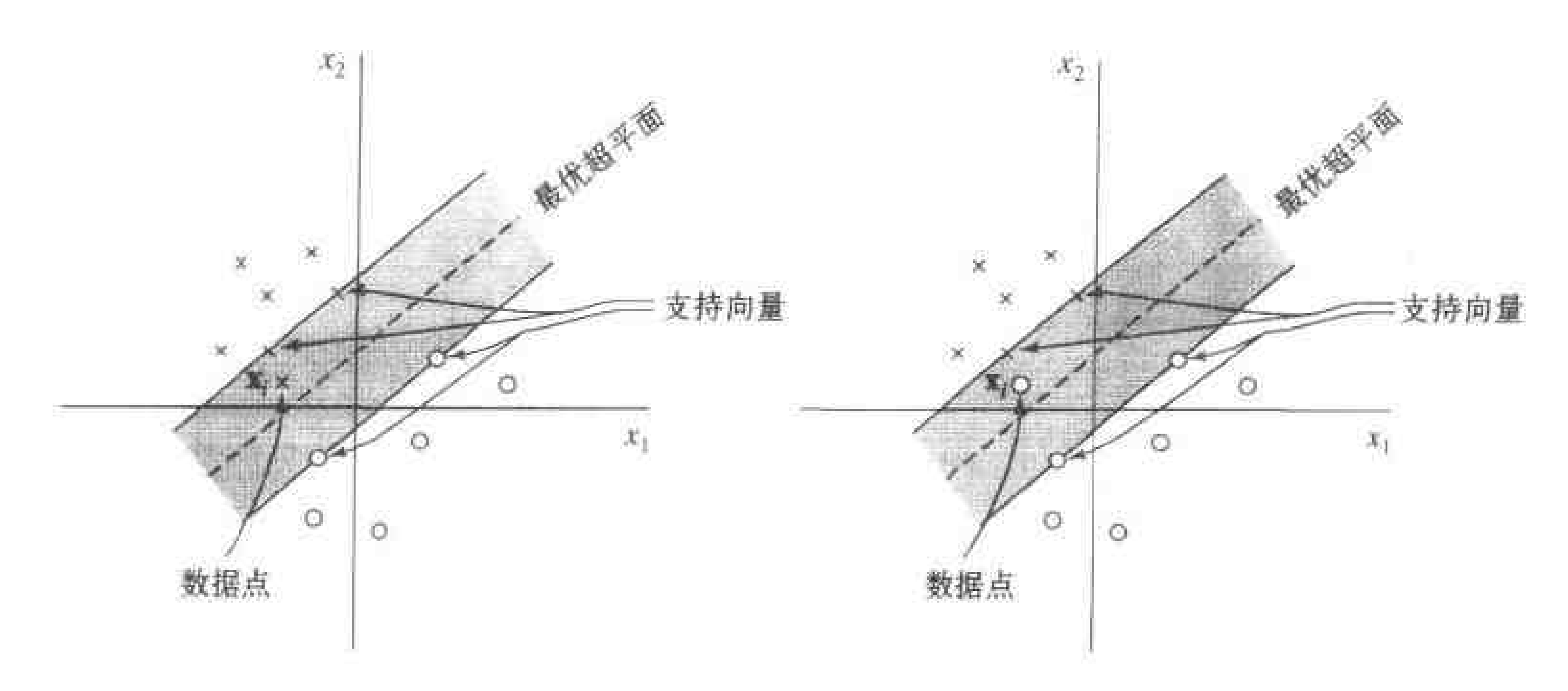


图 6-3

a)数据点 $\mathbf{x}_i$(属于类$\mathscr{C}_1$)落在分离区域之内，但在决策面正确的一侧

b)数据点 $\mathbf{x}_i$(属于类$\mathscr{C}_2$)落在决策面错误的一侧

326

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1,2,\cdots,N \tag{6.22}$$

这里 ξ_i 称为松弛变量(slack variable)；它们度量一个数据点对模式可分的理想条件的偏离程度。对于 $0 \leq \xi_i \leq 1$，数据点落入分离区域的内部，但是在决策面的正确一侧，如图 6-3a 所示。对于 $\xi_i > 1$，数据点落到分离超平面的错误一侧，如图 6-3b 所示。支持向量是那些精确满足式(6.22)的特殊数据点，即使 $\xi_i > 0$。注意，如果一个 $\xi_i > 0$ 对应的样本被遗弃在训练集外，决策面就要改变。因此，支持向量的定义对线性可分和不可分的情况都是相同的。

我们的目标是找到分离超平面使其在训练集上的平均错误分类的误差最小。为了达到这一点，通过对权值向量 $\mathbf{w}$ 最小化泛函

$$\Phi(\xi) = \sum_{i=1}^{N} I(\xi_i - 1)$$

泛函满足式(6.22)的约束条件和对 $\|\mathbf{w}\|^2$ 的限制。函数 $I(\xi)$是一个指标函数，由

$$I(\xi) = \begin{cases} 0 & 若\ \xi \leq 0 \\ 1 & 若\ \xi > 0 \end{cases}$$

定义。不幸的是，$\Phi(\xi)$对 $\mathbf{w}$ 的最小化是非凸的最优化问题，它是 NP－完全的[2]。

为了使最优化问题数学上易解，我们写出

$$\Phi(\xi) = \sum_{i=1}^{N} \xi_i$$

逼近泛函 $\Phi(\xi)$。而且，通过形成泛函对权值向量 $\mathbf{w}$ 的最小化公式简化计算，即

$$\Phi(\mathbf{w},\xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \tag{6.23}$$

如前一样，最小化式(6.23)中第 1 项与最小化支持向量机的 VC 维数有关。至于第 2 项 $\sum_i \xi_i$，它是测试错误数目的一个上界。在式(6.23)中代价函数的公式与结构风险最小化原则完全吻合。

参数 C 控制机器的复杂性和不可分离点数之间的平衡；这样它也可以被看作是一个“正则化”参数的形式。参数 C 由使用者选定。这可由下面两种方法之一完成：

- 参数 C 由实验决定，通过标准使用训练/(确认)测试集，它是重采样的粗略形式。
- 它由分析决定，从式(6.19)估计 VC 维数和使用基于 VC 维数的机器泛化性能的界。

无论哪种情况，泛函 $\Phi(\mathbf{w},\xi)$ 对 $\mathbf{w}$ 和 $\{\xi_i\}_{i=1}^N$ 求最优化，满足式(6.22)描述的约束条件和 $\xi_i \geqslant 0$。这样做，$\mathbf{w}$ 的范数平方被认为是一个关于不可分离点的联合最小化中一个数量项，而不是作为强加在关于不可分离点数量的最小化上的一个约束条件。

对刚刚陈述的不可分模式的最优化问题，包括线性可分模式的最优化问题作为它的一种特殊情况。具体地，在式(6.22)和式(6.23)中对所有的 i 置 $\xi_i = 0$，就把它们化简为相应的线性可分情形。

327

我们现在对不可分离的情况的原问题可以正式地陈述如下：

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$，寻找权值向量 $\mathbf{w}$ 和偏置 b 的最优值，使得它们满足约束条件

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 - \xi_i \qquad 对于\ i = 1,2,\cdots,N$$

$$\xi_i \geqslant 0 \qquad 对所有的\ i$$

并且使得权值向量 $\mathbf{w}$ 和松弛变量 ξ_i 最小化代价函数

$$\Phi(\mathbf{w},\xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^N \xi_i$$

其中，C 是使用者选定的正参数。

使用 Lagrange 乘子的方法，以一种与6.2节所描述的相似方式来处理，我们可以得到不可分离模式的对偶问题的表示如下(参看习题6.3)：

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$，寻找最大化目标函数

$$Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N \alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

的 Lagrange 乘子 $\{\alpha_i\}_{i=1}^N$，满足约束条件

(1) $\sum_{i=1}^N \alpha_i d_i = 0$,

(2) $0 \leqslant \alpha_i \leqslant C \qquad 对于\ i = 1,2,\cdots,N$

其中，C 是使用者选定的正参数。

注意，松弛变量 ξ_i 和它们的 Lagrange 乘子都不出现在对偶问题里。除了一些少许的但很重要的差别外，不可分模式的对偶问题与线性可分模式的简单情况相似。在两种情况下，最大化的目标函数 $Q(\alpha)$ 是相同的。不可分离情况与可分离情况的不同在于限制条件 $\alpha_i \geqslant 0$ 被替换为条件更强的 $0 \leqslant \alpha_i \leqslant C$。除了这个修改，不可分离情况的约束最优化问题和权值向量 $\mathbf{w}$ 和偏置 b 的最优值计算过程与线性可分离情况的一样。还要注意支持向量和以前的定义相同。

权值向量 $\mathbf{w}$ 的最优解由

$$\mathbf{w}_o = \sum_{i=1}^{N_S} \alpha_{o,i} d_i \mathbf{x}_i \tag{6.24}$$

328

给出，其中 N_S 是支持向量的个数。决定偏置最优值所使用的方法也与以前描述的过程相似。具体地，Kuhn-Tucker 条件被定义为

$$\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i] = 0, \qquad i = 1,2,\cdots,N \tag{6.25}$$

和

$$\mu_i\xi_i = 0, \qquad i = 1,2,\cdots,N \tag{6.26}$$

式(6.25)是式(6.14)的改写，单位1被$(1-\xi_i)$代替。至于式(6.26)，μ_i 是 Lagrange 乘子，引入它的目的是对所有 i 强制松弛变量 ξ_i 为非负。在鞍点对于原问题的 Lagrange 函数对松弛变量 ξ_i 的导数的值为零，计算这个值得到

$$\alpha_i + \mu_i = C \tag{6.27}$$

联合式(6.26)和式(6.27)，我们有

$$\xi_i = 0, \quad 如果 \quad \alpha_i < C \tag{6.28}$$

我们可以决定最优偏置量 b_o 如下，取训练集中满足 $0 < \alpha_{o,i} < C$ 的任意数据点$(\mathbf{x}_i, d_i)$，因此 $\xi_i = 0$，并对式(6.25)使用那个数据点。然而，从数值的角度看，采用从训练样本中所有这样的数据点得到的 b_o 的平均值更好(Burges, 1998)。

6.4 怎样建立用于模式识别的支持向量机

有了关于对不可分离模式怎样找到最优超平面的知识，我们现在正式描述建立用于模式－识别任务的支持向量机。

基本上，支持向量机[13] 的思想建立在两个数学运算上，现概述如下并在图6-4中说明：

1. 输入向量到高维特征空间的非线性映射，特征空间对输入和输出都是隐藏的。
2. 构造一个最优超平面用于分离在第1步中发现的特征。

两个操作的基本理由在下面解释。

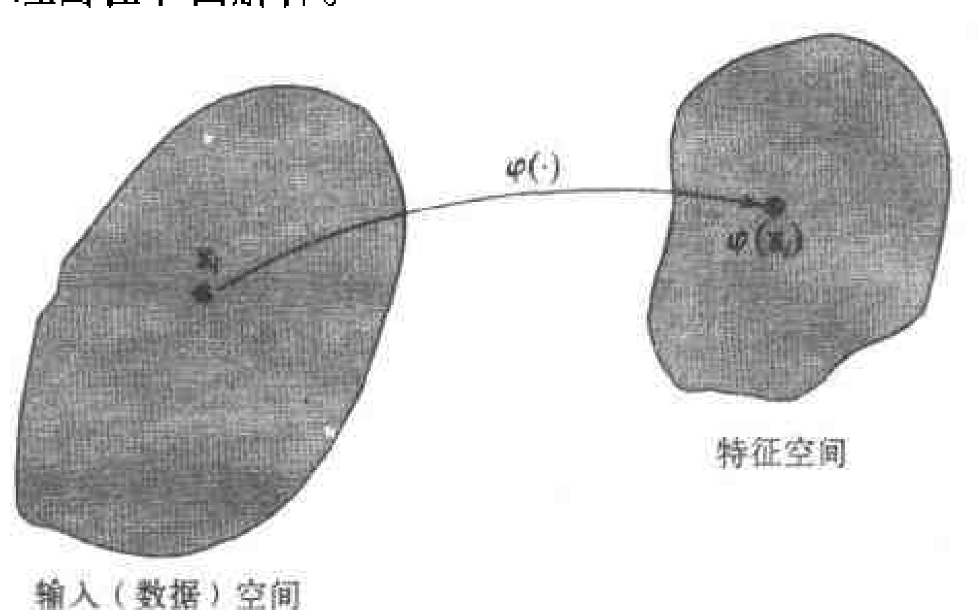


图6-4 从输入空间到特征空间的非线性映射 φ(·)

329

操作1根据第5章中讨论的关于模式可分性的 Cover 定理执行。考虑由非线性可分模式构成的输入空间。Cover 定理陈述为：如果两个条件均满足，那么多维空间能变换为一个新的特征空间，使得在特征空间中模式以较高的概率为线性可分的。首先，变换是非线性的。其次，特征空间的维数是足够高的。这两个条件在操作1中体现。然而，注意 Cover 定理没有讨论分离超平面的最优性。只有使用一个最优分离超平面使 VC 维数达到最小和获得泛化能力。

接着要说明的是第2个操作从何而来。具体地，操作2利用建立最优分离超平面的思想，它根据6.3节描述的理论，但是有一个根本的不同：现在分离超平面被定义为从特征空

间得出的向量线性函数，而不是从原始输入空间。更重要的是，这个超平面的构造与建立在VC维数理论上的结构风险最小化的原则是一致的。这个构造与内积核的求值有关。

内积核

令 $\mathbf{x}$ 表示从输入空间得到的向量，假定维数为 m_0。令 $\{\varphi_j(\mathbf{x})\}_{j=1}^{m_1}$ 表示从输入空间到特征空间的一个非线性变换的集合；m_1 是特征空间的维数。对所有的 j，假定 $\varphi_j(\mathbf{x})$ 根据先验知识定义的。给定非线性变换的这样一个集合，可以定义一个充当决策面的超平面

$$\sum_{j=1}^{m_1} w_j \varphi_j(\mathbf{x}) + b = 0 \tag{6.29}$$

其中 $\{w_j\}_{j=1}^{m_1}$ 表示把特征空间连接到输出空间的线性权值的集合，b 是偏置。我们可以简化为

$$\sum_{j=0}^{m_1} w_j \varphi_j(\mathbf{x}) = 0 \tag{6.30}$$

其中假定对所有的 $\mathbf{x}$，$\varphi_0(\mathbf{x})=1$，所以 w_0 表示偏置 b。式(6.30)定义了一个决策面，这个决策面在特征空间根据机器的线性权值进行计算。通过特征空间，$\varphi_j(\mathbf{x})$ 表示提供给权值 w_j 的输入。定义向量

$$\boldsymbol{\varphi}(\mathbf{x}) = [\varphi_0(\mathbf{x}), \varphi_1(\mathbf{x}), \cdots, \varphi_{m_1}(\mathbf{x})]^T \tag{6.31}$$

其中，由定义有

$$\varphi_0(\mathbf{x}) = 1 \qquad \text{对所有的 } \mathbf{x} \tag{6.32}$$

实际上，向量 $\boldsymbol{\varphi}(\mathbf{x})$ 表示由于输入向量 $\mathbf{x}$ 在特征空间诱导出的“像”，如图 6-4 所示。那么，利用这个像用紧凑的形式定义决策面：

330

$$\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = 0 \tag{6.33}$$

我们使式(6.12)适合现在涉及特征空间的情形，在特征空间中现在寻找特征的“线性”可分性，可以写成

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \boldsymbol{\varphi}(\mathbf{x}_i) \tag{6.34}$$

其中特征向量 $\boldsymbol{\varphi}(\mathbf{x}_i)$ 与在第 i 个例子里输入模式 $\mathbf{x}_i$ 相对应。因此将式(6.34)代入式(6.33)，可以定义在特征空间中计算的决策面如下：

$$\sum_{i=1}^{N} \alpha_i d_i \boldsymbol{\varphi}^T(\mathbf{x}_i) \boldsymbol{\varphi}(\mathbf{x}) = 0 \tag{6.35}$$

项 $\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x})$ 表示特征空间中由第 i 个例子的输入模式 $\mathbf{x}_i$ 和输入向量 $\mathbf{x}$ 诱导的两个向量内积。这样我们可以引入内积核(inner-product kernel)，由 $K(\mathbf{x},\mathbf{x}_i)$ 表示并且定义为

$$K(\mathbf{x},\mathbf{x}_i) = \boldsymbol{\varphi}^T(\mathbf{x})\boldsymbol{\varphi}(\mathbf{x}_i) = \sum_{j=0}^{m_1} \varphi_j(\mathbf{x})\varphi_j(\mathbf{x}_i) \qquad i = 1,2,\cdots,N \tag{6.36}$$

从这个定义，立即看出内积核是自变量的对称函数，表示为

$$K(\mathbf{x},\mathbf{x}_i) = K(\mathbf{x}_i,\mathbf{x}) \qquad \text{对所有的 } i \tag{6.37}$$

最重要的是，我们可以使用内积核 $K(\mathbf{x},\mathbf{x}_i)$ 在特征空间中建立最优超平面，无需用显式的形式考虑特征空间自身。将式(6.36)代入(6.35)容易看出这一点，此时最优超平面定义为

$$\sum_{i=1}^{N}\alpha_i d_i K(\mathbf{x},\mathbf{x}_i) = 0 \tag{6.38}$$

Mercer 定理

式(6.36)对于内积核函数 $K(\mathbf{x},\mathbf{x}_i)$的展开是在泛函分析中出现的 Mercer 定理的一种特殊情形。这个定理可以正式表述如下(Mercer,1908;Courant and Hilbert,1970):

$K(\mathbf{x},\mathbf{x}')$表示一个连续的对称核，其中 $\mathbf{x}$ 定义在闭区间 $\mathbf{a}\leqslant\mathbf{x}\leqslant\mathbf{b}$,$\mathbf{x}'$类似。核 $K(\mathbf{x},\mathbf{x}')$可以被展开为级数

$$K(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{\infty}\lambda_i\varphi_i(\mathbf{x})\varphi_i(\mathbf{x}') \tag{6.39}$$

其中所有的 λ_i 均是正的。为了保证这个展开式是合理的并且为绝对一致收敛的，充要条件是条件

$$\int_{\mathbf{b}}^{\mathbf{a}}\int_{\mathbf{b}}^{\mathbf{a}}K(\mathbf{x},\mathbf{x}')\Psi(\mathbf{x})\Psi(\mathbf{x}')\,d\mathbf{x}d\mathbf{x}' \geqslant 0$$

331

对于所有满足 $\int_{\mathbf{b}}^{\mathbf{a}}\Psi^2(\mathbf{x})\,d\mathbf{x} < \infty$ 的 $\Psi(\cdot)$ 成立。

函数 $\varphi_i(\mathbf{x})$称为展开的*特征函数*，λ_i 称为*特征值*。所有的特征值均为正数这个事实意味着核 $K(\mathbf{x},\mathbf{x}')$是*正定的*。

根据 Mercer 定理，我们有如下的结论：

- 对于 $\lambda_i\neq1$，输入向量 $\mathbf{x}$ 在特征空间中诱导出的第 i 个像$\sqrt{\lambda_i}\varphi_i(\mathbf{x})$是一个展开的特征函数。
- 理论上，特征空间的维数(即特征值/特征函数的数目)可以是无穷大。

Mercer 定理仅告诉我们一个候选核是不是一个在某个空间中的内积核，从而允许用于一个支持向量机。但是，它并没有说如何去构造函数 $\varphi_i(\mathbf{x})$；我们不得不自己来做。

从定义式(6.23)可以看出，支持向量机包含一种*隐含的*正则化形式。特别地，使用根据 Mercer 定理定义的核 $K(\mathbf{x},\mathbf{x}')$和根据算子 $\mathbf{D}$ 进行正则化对应，使得核函数 $K(\mathbf{x},\mathbf{x}')$是 $\tilde{\mathbf{D}}\mathbf{D}$ 的格林函数，其中 $\tilde{\mathbf{D}}$ 是 $\mathbf{D}$ 的伴随算子(Smola and Schölkopf,1998)。正则化理论在第 5 章讨论。

支持向量机的最优设计

式(6.36)的内积核 $K(\mathbf{x},\mathbf{x}_i)$的展开式允许我们建立一个决策面，在输入空间中它是非线性的，但它在特征空间的像是线性的。有了这个展开式，我们现对支持向量机受约束的最优化的对偶形式陈述如下：

给定训练样本$\{(\mathbf{x}_i,d_i)\}_{i=1}^{N}$，寻找最大化目标函数

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j K(\mathbf{x}_i,\mathbf{x}_j) \tag{6.40}$$

的 Lagrange 乘子$\{\alpha_i\}_{i=1}^{N}$，满足约束条件

(1) $\sum_{i=1}^{N}\alpha_i d_i = 0$

(2) $0 \leqslant \alpha_i \leqslant C \qquad i = 1,2,\cdots,N$

其中，C 是使用者选定的正参数。

注意，约束(1)由 Lagrange 函数 $Q\{\alpha\}$ 对 $\varphi_0(\mathbf{x}) = 1$ 对应的偏置 $b = w_0$ 的最优化产生。这里陈述的对偶问题与在 6.3 节中考虑的不可分模式情况的形式相同，除了内积 $\mathbf{x}_i^T\mathbf{x}_j$ 被内积核 $K(\mathbf{x}_i,\mathbf{x}_j)$代替的事实。我们可以把 $K(\mathbf{x}_i,\mathbf{x}_j)$看作是 $N\times N$ 的对称矩阵 $\mathbf{K}$ 的第 ij - 项元素，表示为

332

$$\mathbf{K} = \{K(\mathbf{x}_i,\mathbf{x}_j)\}_{(i,j)=1}^{N} \tag{6.41}$$

在找到了由 $\alpha_{o,i}$表示的 Lagrange 乘子的最优值之后，我们可以确定相应的线性权值向量最优值 $\mathbf{w}_o$，在新的情况下它改变式(6.17)的公式联系特征空间到输出空间。特别地，认识到像 $\varphi(\mathbf{x}_i)$从输入到权值向量 $\mathbf{w}$ 所起的作用，我们可以定义 $\mathbf{w}_o$ 为

$$\mathbf{w}_o = \sum_{i=1}^{N}\alpha_{o,i}d_i\boldsymbol{\varphi}(\mathbf{x}_i) \tag{6.42}$$

其中 $\boldsymbol{\varphi}(\mathbf{x}_i)$是 $\mathbf{x}_i$ 在特征空间诱导的像。注意 $\mathbf{w}_o$ 的第一个分量表示最优偏置 b_o。

支持向量机的例子

核 $K(\mathbf{x},\mathbf{x}_i)$的要求是满足 Mercer 定理。在这个要求之内，怎样选择它是有一定自由度的。表 6-1 小结支持向量机的三个普遍类型的内积核函数：多项式学习机器，径向基函数网络，两层感知器。下面几点是值注意的：

表 6-1 内积核小结

333

支持向量机类型	内积核 $K(\mathbf{x},\mathbf{x}_i)$, $i=1,2,\cdots,N$	评　述
多项式学习机	$(\mathbf{x}^T\mathbf{x}_i + 1)^p$	指数 p 由使用者预先指定
径向基函数网络	$\exp\left(-\frac{1}{2\sigma^2}\lVert\mathbf{x}-\mathbf{x}_i\rVert^2\right)$	宽度 σ^2 对所有核相同，由使用者预先指定
两层感知器	$\tanh(\beta_0\mathbf{x}^T\mathbf{x}_i + \beta_1)$	只有一些特定的 β_0，β_1 值满足 Mercer 定理

1. 用于支持向量机的多项式和径向基函数类型的内积核总满足 Mercer 定理。相反，用于支持向量机的两层感知器的类型，其内积核受到某种限制，如表 6-1 最后一行所示。后面的条目证实如下的事实：判定一个给定的核是否符合 Mercer 定理确实是一件困难的事情；见习题 6.8。

2. 对所有三种机器类型，特征空间维数由从训练数据抽取的支持向量的个数决定，这些训练数据是通过解决受约束最优化问题来获得的。

3. 支持向量机的基本理论避免启发式的需要，它们常被用在传统的径向基函数网络和多层感知器的设计上面：

- 在径向基函数类型的支持向量机中，径向基函数的数量和它们的中心分别由支持向量的个数和支持向量的值自动决定。
- 在两层感知器类型的支持向量机中，隐藏神经元的个数和它们的权值向量分别由支持向量的个数和支持向量的值自动决定。

图6-5显示一个支持向量机的体系结构。

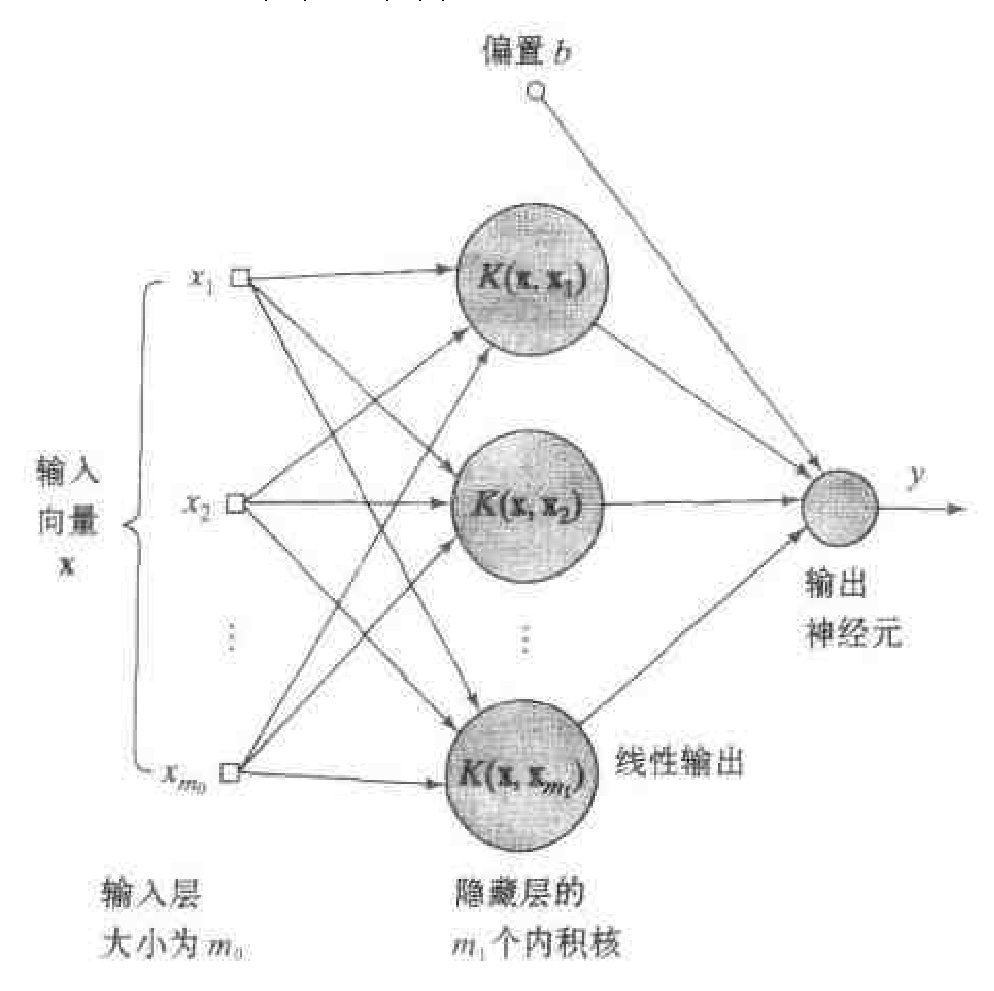


图6-5 支持向量机的体系结构

不管支持向量机是怎样实现的，基本上它是不同于传统的设计多层感知器的方法。在传统的方法里，模型复杂性由保持特征(即隐藏神经元)的数量最小所控制。另一方面，支持向量机提供一个学习机器设计的解决方案，其模型复杂性的控制独立于维数，小结如下(Vapnik,1995,1998)：

- **概念问题**。有意使特征(隐藏)空间的维数足够大，使得可以在这个空间建立超平面形式的决策面。为了一个好的泛化性能，模型的复杂性通过对所建立的超平面添加一些特定的约束条件来控制，这导致训练数据中的一小部分被抽出来作为支持向量。
- **计算问题**。在高维空间的数值最优化受到维数灾的影响。通过使用一个内积核(按照Mercer定理定义)的概念，和求解在输入(数据)空间用形成的约束最优化问题的对偶形式，避免计算上的问题。 334

6.5 例子：XOR问题(再讨论)

为了说明支持向量机设计过程，我们再讨论在第4章和第5章讨论过的XOR(异或)问题。表6-2给出了4个可能状态的输入向量和期望的响应。

表6-2 XOR问题

输入向量 $\mathbf{x}$	期望响应 d
(-1,-1)	-1
(-1,+1)	+1
(+1,-1)	+1
(+1,+1)	-1

为了进行处理，令(Cherkassky and Mulier,1998)

$$K(\mathbf{x},\mathbf{x}_i) = (1 + \mathbf{x}^T\mathbf{x}_i)^2 \tag{6.43}$$

用 $\mathbf{x} = [x_1, x_2]^T$ 和 $\mathbf{x}_i = [x_{i1}, x_{i2}]^T$，因而内积核 $K(\mathbf{x},\mathbf{x}_i)$可应用不同次数的单项式表示如下：

$$K(\mathbf{x},\mathbf{x}_i) = 1 + x_1^2 x_{i1}^2 + 2x_1 x_2 x_{i1} x_{i2} + x_2^2 x_{i2}^2 + 2x_1 x_{i1} + 2x_2 x_{i2}$$

输入向量 $\mathbf{x}$ 在特征空间中诱导的像可推断为

$$\boldsymbol{\varphi}(\mathbf{x}) = [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$$

类似地 $\boldsymbol{\varphi}(\mathbf{x}_i) = [1, x_{i1}^2, \sqrt{2}x_{i1} x_{i2}, x_{i2}^2, \sqrt{2}x_{i1}, \sqrt{2}x_{i2}]^T, \quad i = 1,2,3,4$

由式(6.41)我们可发现

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$$

因此目标函数的对偶形式为(参看式(6.40))：

$$Q(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}(9\alpha_1^2 - 2\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 2\alpha_1\alpha_4 +$$
$$9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2)$$

对 Lagrange 乘子优化 $Q(\alpha)$产生下列联立方程组：

$$\begin{aligned} 9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 &= 1 \\ -\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 &= 1 \\ -\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 &= 1 \\ \alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 &= 1 \end{aligned}$$

335

因此，Lagrange 乘子的最优值为

$$\alpha_{o,1} = \alpha_{o,2} = \alpha_{o,3} = \alpha_{o,4} = \frac{1}{8}$$

这个结果说明，本例中所有 4 个输入向量$\{\mathbf{x}_i\}_{i=1}^4$都是支持向量。$Q(\alpha)$的最优值是

$$Q_o(\alpha) = \frac{1}{4}$$

相应地，我们可写出

$$\frac{1}{2}\|\mathbf{w}_o\|^2 = \frac{1}{4}$$

或

$$\|\mathbf{w}_o\| = \frac{1}{\sqrt{2}}$$

从式(6.42)，我们发现最优权值向量是

$$\mathbf{w}_o = \frac{1}{8}[-\varphi(\mathbf{x}_1) + \varphi(\mathbf{x}_2) + \varphi(\mathbf{x}_3) - \varphi(\mathbf{x}_4)]$$

$$= \frac{1}{8}\left[-\begin{bmatrix} 1 \\ 1 \\ \sqrt{2} \\ 1 \\ -\sqrt{2} \\ -\sqrt{2} \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -\sqrt{2} \\ 1 \\ -\sqrt{2} \\ \sqrt{2} \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -\sqrt{2} \\ 1 \\ \sqrt{2} \\ -\sqrt{2} \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \sqrt{2} \\ 1 \\ \sqrt{2} \\ \sqrt{2} \end{bmatrix}\right] = \begin{bmatrix} 0 \\ 0 \\ -1/\sqrt{2} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\mathbf{w}_o$ 的第一个分量表示偏置 b 为 0。

最优超平面定义为(参看式(6.33))

$$\mathbf{w}_o^T \boldsymbol{\varphi}(\mathbf{x}) = 0$$

336

即

$$\left[0,\ 0,\ \frac{-1}{\sqrt{2}},\ 0,\ 0,\ 0\right]\begin{bmatrix}1\\ x_1^2\\ \sqrt{2}x_1x_2\\ x_2^2\\ \sqrt{2}x_1\\ \sqrt{2}x_2\end{bmatrix}=0$$

这归结为

$$-x_1x_2=0$$

对于 XOR 问题的多项式形式的支持向量机见图 6-6a。对 $x_1=x_2=-1$ 和 $x_1=x_2=+1$，输出 $y=-1$；对 $x_1=-1$，$x_2=+1$ 以及 $x_1=+1$，$x_2=-1$，输出 $y=+1$。因此如图 6-6b 所示，XOR 问题获得解。

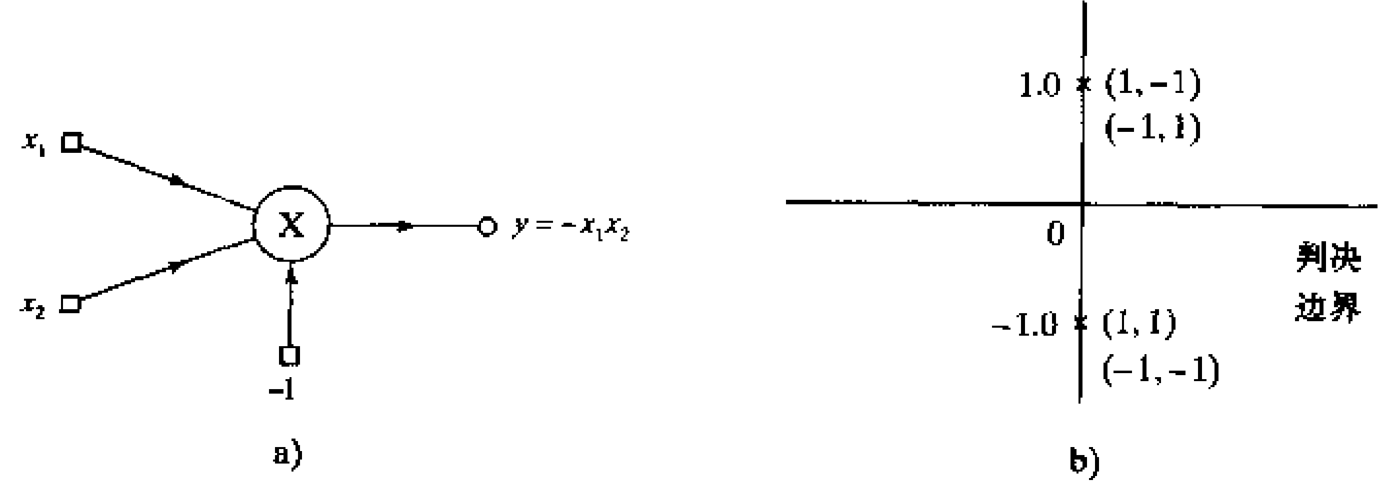


图 6-6

a)解决 XOR 问题的多项式机器 b)XOR 问题的四个数据点在特定空间导出的映像

6.6 计算机实验

在这个计算机实验中，我们回到第 4 章和第 5 章研究过的模式识别问题。实验涉及两个部分重叠的标记为 1($\mathscr{C}_1$ 类)和标记为 2($\mathscr{C}_2$ 类)二维高斯分布的分类。这两个数据集的散列图可以见图 4-14。用 Bayes(最优)分类器所得到的正确分类的概率为

$$p_c = 81.15\%$$

337

表 6-3 给出利用支持向量机对这个数据集进行计算机实验所获得的结果的小结。对于内积核，我们使用了径向基函数

$$K(\mathbf{x},\mathbf{x}_i) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right), \qquad i=1,2,\cdots,N$$

其中相同的宽度 $\sigma^2=4$ 被用于数据集中所有的点。机器对总数为 $N=500$ 的数据点上进行训练，这些数据点是从代表这两个类的数据的总体中随机抽取的。用于正则化的参数 $C=0.1$。

表 6-3 给出的结果是从 5 次不同的实验中得到的，对于每次试验，都采用 500 个点进行训练，并用 32 000 个数据点进行测试。这五次试验的平均正确分类的概率是 81.40%，这个平均值几乎和从 Bayes 分类器得到的相等。在这些实验的一次实验中，最优结果被超出了

0.05%，这是由于试验误差产生的。

表 6-3 使用支持向量机的两类模式分类试验结果小结

	共同宽度 $\sigma^2=4$，正则化参数 $C=0.1$				
正确分类的概率 p_c	81.22	81.28	81.55	81.49	81.45
支持向量数目 N_S	298	287	283	287	286

这个由支持向量机获得的近乎完美的分类结果由图 6-7 所示的决策边界进一步确定，这个图是由这五次机器的实现中随机挑出的一次得到的。在这个图中 Bayes 分类器的决策边界也包括在内，边界是由一个圆构成的，圆心是 $\mathbf{x}_c=[-2/3,0]^T$，半径是 $r=2.34$。图 6-6 清楚显示支持向量机可以构造类$\mathscr{C}_1$ 和类$\mathscr{C}_2$ 间的决策边界使得它几乎和最优决策边界相同。

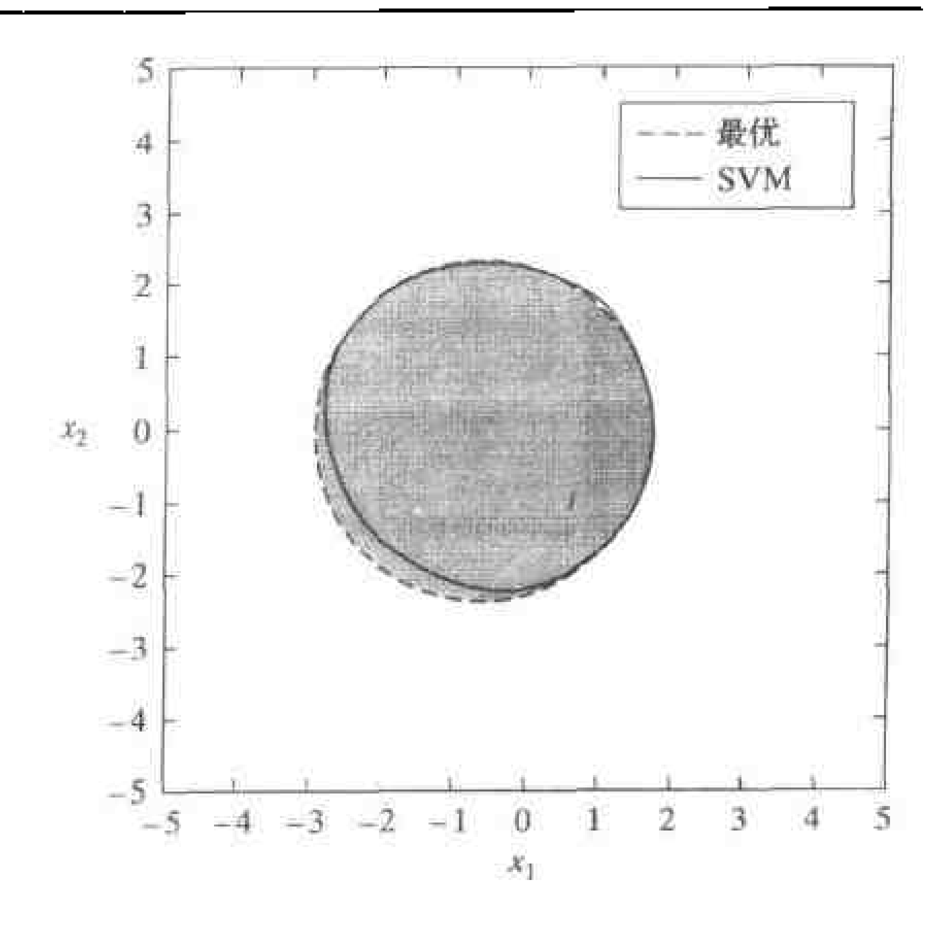


图 6-7 模式分类计算机实验的决策面

让我们回到表 6-3 给出的实验结果的小结，第二行显示支持向量机的 5 个不同实现的大小。这些结果表示对于这个试验，支持向量机学习算法选择了将近 60% 的数据点作为支持向量。

对于不可分离的模式，所有训练误差导致它们自身的支持向量，这是从 Kuhn-Tucker 条件得到的。对于目前的实验，误差率约为 20%。对于一个大小为 500 的样本，我们发现大约 1/3 的支持向量事实上是由于分类误差而产生的。

简评

比较这个建立在支持向量机基础上的简单计算机实验的结果，和 4.8 节报告的在多层感知器上对同一个数据样本采用误差反向传播算法进行训练产生的相应结果，我们可以得出以下结论：

338

1. 对于感兴趣的问题，支持向量机具有*以接近最优的方式*解决模式分类问题的固有能力。此外，它能获得如此显著的性能而无需在机器的设计中嵌入问题域知识。

2. 另一方面，利用反向传播算法训练的多层感知器提供模式分类问题的*计算高效的解*。对这里描述的两类实验，我们能够利用仅用两个隐藏神经元的多层感知器达到 79.70% 的正确分类概率。

在做这个简评中，我们突出了模式分类的这两种方法各自的优点。但是，为了得到公允的评论我们必须确认它们各自的缺点。在支持向量机的情况，近乎完美的分类性能是付出很大计算复杂性代价而取得的。另一方面，对利用反向传播算法训练多层感知器对同一模式分类任务，要达到和支持向量机差不多的性能，必须做两件事：在感知器的设计中建立问题领域的知识，以及调整大量的设计参数，对于困难学习任务这是令人头痛的实践。

6.7 ε－不敏感损失函数

到目前为止，本章集中于利用支持向量机求解模式识别任务。现在，我们考虑利用支持向量机求解非线性回归问题。为了准备这个讨论，我们首先讨论适合这类学习任务的最优化准则问题。

339

在第4章关于多层感知器和第5章关于径向基函数网络的讨论中，我们利用二次损失函数作为这些网络的优化准则。利用这个准则的主要原因是数学上的，即为了计算上的方便。但是，最小二乘估计器对异常点(outlier)(即对于一个微小模型得到异常大的观察)的出现非常敏感，并且当加性噪声的固有分布有很长的尾部时它表现很差。为了克服这些局限，我们需要一种鲁棒的估计器，它对模型小的改变不敏感。

以鲁棒性作为设计目标，对于任何鲁棒性的数值度量必须考虑到由于微小噪声模型的一个ε－偏差而可能产生最大性能退化。根据这种观点，一种最优鲁棒估计过程是最小化最大的性能恶化，因而是一种最小最大过程(Huber,1981)。当加性噪声的概率密度函数关于原点对称时，求解非线性回归问题的最小最大过程[4]利用绝对误差作为被最小化的量(Huber, 1964)。也就是说，损失函数具有形式

$$L(d,y)=|d-y| \tag{6.44}$$

其中 d 是期望响应而 y 是估计器输出

为了构造支持向量机逼近期望的响应 d，我们利用式(6.44)的损失函数的扩展，它由Vapnik(1995,1998)最早提出，这里可描述为

$$L_\varepsilon(d,y)=\begin{cases}|d-y|-\varepsilon, & \text{对于 } |d-y|\geqslant\varepsilon\\ 0 & \text{其他}\end{cases} \tag{6.45}$$

其中ε是指定的参数，损失函数 $L_\varepsilon(d,y)$ 称为ε－不敏感损失函数(ε-insensitive loss function)。如果估计器输出 y 和期望输出 d 的偏差的绝对值小于ε，则它等于零，否则它等于偏差绝对值减去ε。式(6.44)的损失函数是ε－不敏感损失函数在ε＝0时的特殊情形，图6-8说明 $L_\varepsilon(d,y)$ 和误差 $d-y$ 的依赖关系。

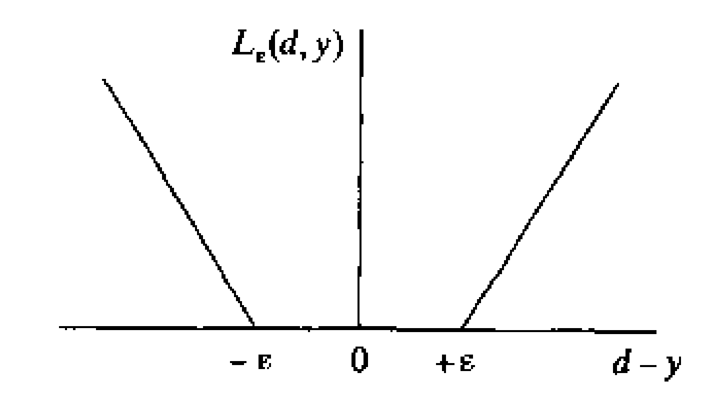


图6-8 ε－不敏感损失函数

6.8 用于非线性回归的支持向量机

考虑非线性回归模型，标量 d 对向量 $\mathbf{x}$ 的依赖可描述为

340

$$d=f(\mathbf{x})+v \tag{6.46}$$

标量值非线性函数 $f(\mathbf{x})$ 定义为在第2章讨论的条件期 $E[D|\mathbf{x}]$；D 是一个随机变量，它的一次实现记为 d。加性噪声项 v 是统计独立于输入向量 $\mathbf{x}$ 的，函数 $f(\cdot)$ 和噪声 v 的统计特性是未知的。我们所有可用的信息就是一组训练数据 $\{(\mathbf{x}_i,d_i)\}_{i=1}^N$，其中 $\mathbf{x}_i$ 是输入向量 $\mathbf{x}$ 的一个样本值，d_i 是模型输出 d 的相应值。问题是提供 d 对 $\mathbf{x}$ 的依赖的估计。

进一步我们假设 d 的估计记为 y，它是由一组非线性基函数 $\{\varphi_j(\mathbf{x})\}_{j=0}^{m_1}$ 的展开得到的：

$$y=\sum_{j=0}^{m_1}w_j\varphi_j(\mathbf{x})=\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}) \tag{6.47}$$

其中 $$\boldsymbol{\varphi}(\mathbf{x})=[\varphi_0(\mathbf{x}),\varphi_1(\mathbf{x}),\cdots,\varphi_{m_1}(\mathbf{x})]^T$$

和 $$\mathbf{w}=[w_0,w_1,\cdots,w_{m_1}]^T$$

和以前一样假定 $\varphi_0(\mathbf{x})=1$，这样权值 w_0 表示偏置 b。需求解的问题是最小化经验风险

$$R_{\text{emp}}=\frac{1}{N}\sum_{i=1}^{N}L_\varepsilon(d_i,y_i) \tag{6.48}$$

满足不等式

$$\|\mathbf{w}\|^2\leqslant c_0 \tag{6.49}$$

其中 c_0 是常数。ε－不敏感损失函数 $L_\varepsilon(d,y_i)$在前面式(6.45)中定义，我们可以引入两组非负的松弛变量$\{\xi_i\}_{i=1}^N$和$\{\xi'_i\}_{i=1}^N$重新表示这个约束最优化问题，松弛变量定义为：

$$d_i-\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)\leqslant\varepsilon+\xi_i,\qquad i=1,2,\cdots,N \tag{6.50}$$

$$\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)-d_i\leqslant\varepsilon+\xi'_i,\qquad i=1,2,\cdots,N \tag{6.51}$$

$$\xi_i\geqslant0,i=1,2,\cdots,N \tag{6.52}$$

$$\xi'_i\geqslant0,i=1,2,\cdots,N \tag{6.53}$$

松弛变量 ξ_i 和 ξ'_i 描述式(6.45)定义的 ε－不敏感损失函数。因此，这个约束最优化问题等价于最小化代价泛函

341

$$\Phi(\mathbf{w},\xi,\xi')=C\left(\sum_{i=1}^{N}(\xi_i+\xi'_i)\right)+\frac{1}{2}\mathbf{w}^T\mathbf{w} \tag{6.54}$$

满足式(6.50)至(6.53)的约束条件。结合在式(6.54)的泛函 $\Phi(\mathbf{w},\xi,\xi')$中的项 $\mathbf{w}^T\mathbf{w}/2$，我们不需要式(6.49)的不等式约束。在式(6.54)中的常数 C 是用户给定的参数。从而，我们可以定义 Lagrange 函数

$$J(\mathbf{w},\xi,\xi',\alpha,\alpha',\gamma,\gamma')=C\sum_{i=1}^{N}(\xi_i+\xi'_i)+\frac{1}{2}\mathbf{w}^T\mathbf{w}-\sum_{i=1}^{N}\alpha_i[\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)-d_i+\varepsilon+\xi_i]-\sum_{i=1}^{N}\alpha'_i[d_i-\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+\varepsilon+\xi'_i]-\sum_{i=1}^{N}(\gamma_i\xi_i+\gamma'_i\xi'_i) \tag{6.55}$$

其中 α_i 和 α'_i 是 Lagrange 乘子。式(6.55)右边最后一项涉及 γ_i 和 γ'_i 是为了确保 Lagrange 乘子 α_i，α'_i 的最优性条件成为可变形式。要求对 $\mathbf{w}$ 和松弛变量 ξ 和 ξ' 最小化 $J(\mathbf{w},\xi,\xi',\alpha,\alpha',\gamma,\gamma')$；同时也必须对$\alpha_i,\alpha'_i$ 和 γ_i,γ'_i 最大化它。求解这个最优化，我们分别有

$$\mathbf{w}=\sum_{i=1}^{N}(\alpha_i-\alpha'_i)\boldsymbol{\varphi}(\mathbf{x}_i) \tag{6.56}$$

$$\gamma_i=C-\alpha_i \tag{6.57}$$

和

$$\gamma'_i=C-\alpha'_i \tag{6.58}$$

刚才描述的 $J(\mathbf{w},\xi,\xi',\alpha,\alpha',\gamma,\gamma')$最优化是回归的原问题。为了构造相应的对偶问题，我们将(6.56)至(6.58)代入(6.55)中，从而得到凸泛函(经过化简之后)

$$Q(\alpha_i,\alpha'_i)=\sum_{i=1}^{N}d_i(\alpha_i-\alpha'_i)-\varepsilon\sum_{i=1}^{N}(\alpha_i+\alpha'_i)-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i-\alpha'_i)(\alpha_j-\alpha'_j)K(\mathbf{x}_i,\mathbf{x}_j) \tag{6.59}$$

其中 $K(\mathbf{x}_i, \mathbf{x}_j)$是按照 Mercer 定理定义的内积核

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j)$$

我们得到约束最优化问题的解是在对 Lagrange 乘子 α_i 和 α'_i 最大化 $Q(\alpha, \alpha')$得到的，这两个乘子满足加入常数 C 的一组新的约束条件，其中 C 包含在式(6.54)的函数 $\Phi(\mathbf{w}, \xi, \xi')$的定义中。

我们现在可以陈述利用支持向量机的非线性回归的对偶问题如下：

342

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$，寻找 Lagrange 乘子 $\{\alpha_i\}_{i=1}^N$ 和 $\{\alpha'_i\}_{i=1}^N$ 使其最大化目标函数

$$Q(\alpha_i, \alpha'_i) = \sum_{i=1}^N d_i(\alpha_i - \alpha'_i) - \varepsilon\sum_{i=1}^N(\alpha_i + \alpha'_i) - \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N(\alpha_i - \alpha'_i)(\alpha_j - \alpha'_j)K(\mathbf{x}_i, \mathbf{x}_j)$$

满足约束条件

(1) $\sum_{i=1}^N(\alpha_i - \alpha'_i) = 0$

(2) $0 \leqslant \alpha_i \leqslant C, i = 1, 2, \cdots, N$

$0 \leqslant \alpha'_i \leqslant C, i = 1, 2, \cdots, N$

其中 C 为用户给定的常数。

Lagrange 函数最优化问题中，对于 $\varphi_0(\mathbf{x}) = 1$ 的偏置 $b = w_0$ 产生约束条件(1)。因此，获得最优的 α_i 和 α'_i 的值后，对给定的映射 $\varphi(\mathbf{x})$我们可以利用式(6.56)确定权值向量 $\mathbf{w}$ 的最优值。注意和模式识别问题的解一样，在式(6.56)的展开中仅有一些系数不为零；特别，$\alpha_i \neq \alpha'_i$ 对应的数据点定义为机器的支持向量。

ε 和 C 是控制逼近函数

$$F(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x} = \sum_{i=1}^N(\alpha_i - \alpha'_i)K(\mathbf{x}, \mathbf{x}_i) \tag{6.60}$$

VC 维数的自由参数。ε 和 C 两者都必须由用户选择。从概念上讲，ε 和 C 的选择提出和模式分类中参数 C 的选择同样的复杂性控制问题。但是，实际上回归的复杂性控制是一个更困难的问题，这是由于下列原因：

- 参数 ε 和 C 必须同时调整。
- 回归本质上比模式分类更困难。

ε 和 C 选择的原则方法一直是一个未解决的研究领域。

最后，和用于模式识别的支持向量机一样，用于非线性回归的支持向量机可以用多项式项学习机、径向基函数网络或两层感知器实现。三种实现方法的内积核在表 6-1 中给出。

6.9 小结和讨论

支持向量机是为了设计仅含有一个非线性单元隐藏层的前馈网络的一种精巧和高度原则化的学习方法。它由植根于 VC 维理论的结构风险最小化原则导出，这一点使得它的推导更加深奥。正如它的名字所揭示的，机器的设计随抽取训练数据的子集作为支持向量而定，因而代表数据的一个稳定特征。支持向量机包括多项式学习机器、径向基函数网络和两层感知器作为其特殊情形。因此，虽然这些方法提供训练数据的内在统计规则的不同的表示，但是它们都源于支持向量机设置的一个共同基础。

343

与流行的反向传播算法不同，支持向量学习算法仅仅按集中方式进行。这两个算法存在

另一个重要差别。反向传播算法不管学习任务是什么都最小化一个二次损失函数。相反，用于模式识别的支持向量机学习算法和用于非线性回归有很大区别，如下所述：

- 当完成模式识别任务时，支持向量学习算法最小化落在正例和反例分离边缘内的训练样本数目；这只是近似对的，因为使用松弛变量 ξ_i 替代指标函数 $I(\xi_i-1)$。虽然这个准则和最小化分类误差的概率不完全一样，但是它被认为比反向传播学习算法的均方误差准则更适合。
- 当完成非线性回归任务时，支持向量学习算法最小化的 ε - 不敏感损失函数是最小最大理论的平均绝对误差准则的一种推广。因此算法为鲁棒性的。

不管学习任务是什么，支持向量机提供一种独立于维数的控制模型复杂性的方法。特别地，利用定义在特征(隐藏)空间的惩罚超平面作为决策面，模型的复杂性问题在高维空间中得到解决，结果有很好的泛化性能。通过把处理约束最优化问题集中于其对偶问题，绕过维数灾的困难。利用对偶设置的一个重要原因就是避免在数据空间中定义和计算可能的高维数最优超平面的参数。

通常支持向量机的训练包含一个二次规划问题[5]，这个问题由于两个原因而有吸引力：

- 它保证找到误差曲面的全局极值点，在这里误差是指期望响应和支持向量机输出之间的差异。
- 计算可以被有效的执行。

最重要的是，通过使用一个恰当的内积核，支持向量机可以根据内积核的选择自动计算所有重要的网络参数。例如，在径向基函数网络的情形，核函数是 Gauss 函数，对于这种实现方法，径向基函数的数目和它们的中心，以及线性权值和偏置水平，都是自动计算的。径向基函数的中心由二次优化策略挑选的支持向量定义。支持向量通常是由训练样本组成的样本总体的一部分。因此我们可以将利用支持向量机学习过程所得到的 RBF 网络的设计，看作前一章描述的使用严格插值策略得到的设计结果的一种稀疏性版本。

344

可以用几个商用的最优化库[6]求解二次规划问题。但是，这些库的使用受到限制。对于二次规划问题的存储需求随着训练样本的大小平方地增长。从而对现实生活中可能涉及几千个数据点的应用问题，直接利用商用最优化库不能求解二次规划问题。Osuna et al. (1997)已经发展了一种新的分解算法，通过求解一系列更小的子问题取得最优解。特别地，分解算法利用支持向量的系数仅在由 $\alpha_i=0$ 或 $\alpha_i=C$ 定义的边界的一边起作用的这个特点。在那里报告了分解算法能够对具有 100 000 个数据的应用给出满意的结果。

至于运行时间，当前支持向量机在类似的泛化性能上比其他神经网络(例如用反向传播算法训练的多层感知器)慢。有两个原因导致这样慢的行为：

1. 对于由学习算法挑选的用作支持向量的数据点总数目没有控制。
2. 没有预先将任务的先验知识合并到学习机器的设计中。

现在简要讨论为了克服这些缺点而对支持向量机进行的修改。

怎样控制支持向量的选择是一个困难的问题，特别是在待分类的模式为不可分的且训练数据有噪声时。一般地，试图在训练前从数据中消除已知误差或在训练之后从展开中消除它们，将给出不同最优超平面，这是因为惩罚不可分性需要误差。在 Osuna and Girosi (1998)的文章中，研究了减少用于模式识别的支持向量机的训练时间。处理这个问题的两个新方法描述如下：

- 支持向量机用作非线性回归的工具，以用户给定精度逼近决策面(分离不同类)。
- 重新调整训练支持向量机的过程，利用更小数目的基函数产生同样精确的决策面。

在第一种方法中，利用基函数的一个子集的线性组合逼近解，得到的机器是用于函数逼近的支持向量机的自然推广。设计这个推广的目标是寻找下列形式的代价泛函的最小值： 345

$$\mathscr{E}(F) = \sum_{i=1}^{N} |d_i - F(\mathbf{x}_i)|_{\varepsilon} + \frac{1}{2C}\Psi(F)$$

其中 $F(\cdot)$是逼近函数，$\Psi(\cdot)$是光滑度泛函，$|x|_{\varepsilon}$为ε-不敏感代价函数，定义为

$$|x|_{\varepsilon} = \begin{cases} 0 & \text{若 } |x| < \varepsilon \\ |x| - \varepsilon & \text{否则} \end{cases}$$

ε-不敏感代价函数具有使解对奇异点是鲁棒的且对小于阈值ε的误差不敏感的作用。代价泛函$\mathscr{E}(F)$的最小值具有

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i G(\mathbf{x}, \mathbf{x}_i)$$

的形式，其中核 $G(\cdot,\cdot)$依赖于光滑度泛函 $\Psi(\cdot)$的特殊选择，并且通过求解一个二次规划问题计算系数 c_i。解一般是*稀疏的*；那就是，只有少数 c_i 不为零，非零的数目由参数ε控制。

在第二种方法中，原问题被重新表示为和最初的原问题有相同的初始结构，但有一个区别：内积核 $K(\mathbf{x},\mathbf{x}')$结合进新的表示中。这两种方法也适用于减少非线性回归的支持向量机的复杂性。

最后，转到先验知识的问题，人们广泛认识到在机器设计中通过结合任务的先验知识可以提高学习机器的性能(Abu-Mostafa,1995)。一般地，在文献中已经研究两种不同的利用先验知识的方法：

- 在代价函数中包含一个附加项，从而强迫学习机器构造一个加入先验知识的函数。这正是利用正则化所做的事情。
- 从已给训练样本中产生虚拟样本。这里的动机是学习机器从人工扩大的训练集数据中更容易抽取先验知识。

在第二种处理方法中，由于人工数据的相关性和训练数据集的增大，学习过程可能变慢。但是第二种方法比第一种方法有一个优点，那就是对于所有的先验知识和学习机器，它很容易被实现。第二种方法的实现方式可进行如下(Schölkopf et al.,1996)：

1. 按通常方法对给定数据训练支持向量机，抽取一组支持向量。 346
2. 对第1步获得的支持向量，通过以期望的不变性变换形式应用先验知识，生成称为*虚拟支持向量*的人工样本。
3. 对人工增大的样本集训练另一个支持向量机。

这个方法具有以适度的时间代价获得分类精度显著增加的优点：它需要两轮训练而不是一轮训练，但它利用更多的支持向量构造分类规则。

注释和参考文献

[1] 令$\mathscr{C}$为 $\mathbb{R}^m$ 的一个子集，子集$\mathscr{C}$说是凸的，如果

$$\alpha x + (1-\alpha)y \in \mathscr{C} \qquad \text{对所有}(x,y) \in \mathscr{C}\text{和}\ \alpha \in [0,1]$$

函数 $f: \mathscr{C} \to \mathbb{R}$ 说是凸函数，如果

$$f(\alpha x+(1-\alpha)y)\leqslant \alpha f(x)+(1-\alpha)f(y),\text{对所有}(x,y)\in\mathscr{C}\text{和}\alpha\in[0,1]$$

[2] 以计算复杂性作为感兴趣的问题，我们可以确认算法的两种类型：

- *多项式时间算法*，它要求的运行时间是问题大小的多项式函数。例如，通常用于谱分析的快速 Fourier 变换(FFT)算法，是多项式时间算法，它需要运行时间为 $n\log n$，其中 n 为问题的大小。
- *指数时间算法*，它要求运行时间是问题大小的指数函数。例如，一个指数时间算法可能花费时间 2^n，其中 n 为问题大小的度量。

基于此，我们可以将多项式时间算法看作“有效”算法，而指数时间算法看作“无效”算法。

对实际中出现的许多起计算问题，迄今为止仍没有设计出有效算法。如果不是所有的至少也是许多这些看起难解的问题属于称为 NP 完全问题的一类问题。术语“NP”代表“非确定多项式”(Nondeterministic Polynomial)。

关于 NP 完全问题的更详细讨论可参看 Cook(1971)，Garey and Johnson(1979)和 Cormen et al.(1990)。

[3] 在 Aizerman et al.(1964a,1964b)中首次利用内积核的思想构造势函数方法的公式，势函数代表径向基函数网络的前身。几乎在同一时间，Vapnik and Chervonenkis(1965)发展最优超平面的思想。构成支持向量机的这两个有力概念的组合使用是 Vapnik 及合作者 1992 年提倡的；参看 Boser，Guyon and Vapnik(1992)以及 Cortes and Vapnik(1995)。支持向量机的完全数学描述首先在 Vapnik(1995)中给出，随后在 Vapnik(1998) 中以扩展形式给出。

[4] Huber 的最小最大化理论的基础是邻域，这些邻域由于不包含非对称分布，因此不是全局的。但是，这个理论成功解决了一大部分传统的统计学问题，特别的是回归问题。 347

[5] 在 Schurmars(1997)中，利用线性规划探讨使用 L_1 范数 $\|\mathbf{w}\|_1$，替代在支持向量机中使用的 L_2 泛数 $\|\mathbf{w}\|_2$。权值向量 $\mathbf{w}$ 的 L_1 范数定义为

$$\|\mathbf{w}\|_1=\sum_i |w_i|$$

其中 w_i 是 $\mathbf{w}$ 的第 i 个元素，利用 L_1 范数的最大分类边界看上去偏向超平面坐标轴的方向，也就是偏向权值向量具有很少非零元素的方向。

[6] 二次规划的商用库包括下列的软件：

- MINOS5.4:(Murtagh and Saunders,1978)
- LSSOL(Gill et al.,1986)
- LOQO(Vanderbei,1994)
- QPOPT and SQOPT(Gill and Murray,1991)

习题

最优分离超平面

6.1 考虑用于线性可分模式的超平面，它由方程

$$\mathbf{w}^T\mathbf{x}+b=0$$

定义为，其中 $\mathbf{w}$ 表示权值向量，b 为偏置，$\mathbf{x}$ 为输入向量。如果输入模式集 $\{\mathbf{x}_i\}_{i=1}^N$ 满足附加的条件

$$\min_{i=1,2,\cdots,N} | \mathbf{w}^T\mathbf{x}_i + b | = 1$$

则称超平面对应于标准对(canonical pair)($\mathbf{w}$, b)。证明标准对的这个要求导致两类分离边界之间的距离为 $2/\|\mathbf{w}\|$。

6.2 在不可分类模式的背景下判断下列陈述：错分类意味着模式的不可分性，但相反则未必真。

6.3 以不可分模式的分离超平面的最优化作为原问题的开始，构造如6.3节描述的对偶问题的公式。

6.4 在本题中，利用在第4章讨论的"留一法"估计不可分模式的最优超平面产生的期望测试误差。通过删除训练样本中任意一个模式并且根据剩下的模式构造一个解，讨论使用这种方法可以引发的各种可能性。

6.5 数据空间中最优超平面的位置由被选为支持向量的数据点决定。如果数据有噪声，人们的第一反应也许是质疑分离边界对噪声的鲁棒性。但对最优超平面的详细研究揭示分离边界对噪声实际上是鲁棒的。讨论这种鲁棒性的根据。

内积核

6.6 内积核 $K(\mathbf{x}_i,\mathbf{x}_j)$是在训练 N 个样本集$\mathcal{T}$上计算的，它产生 $N\times N$ 矩阵：

$$\mathbf{K} = \{K_{ij}\}_{(i,j)=1}^{N}$$

其中 $K_{ij}=K(\mathbf{x}_i,\mathbf{x}_j)$。由于它的所有元素的值为正，矩阵 $\mathbf{K}$ 是正的。利用相似变换

$$\mathbf{K} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T$$

348

其中 $\boldsymbol{\Lambda}$ 为特征对角矩阵，而 $\mathbf{Q}$ 为相应特征向量构成的矩阵，通过 $\mathbf{K}$ 的特征值和特征向量构造内积核 $K(\mathbf{x}_i,\mathbf{x}_j)$的表达式。你可以从这个表达式得出什么结论？

6.7 (a)证明内积核 $K(\mathbf{x},\mathbf{x}_i)$的酉不变性，即

$$K(\mathbf{x},\mathbf{x}_i) = K(\mathbf{Q}\mathbf{x},\mathbf{Q}\mathbf{x}_i)$$

其中 $\mathbf{Q}$ 为酉矩阵定义为 $\mathbf{Q}^{-1}=\mathbf{Q}^T$

(b)证明表6-1中描述的内积核满足这个性质。

6.8 两层感知器的内积核定义为

$$K(\mathbf{x},\mathbf{x}_i) = \tanh(\beta_0\mathbf{x}^T\mathbf{x}_i + \beta_1)$$

探讨对常数 β_0 和 β_1 的某些值不满足 Mercer 定理。

模式分类

6.9 用于求解 XOR 问题的多项式学习机使用的内积核定义为

$$K(\mathbf{x},\mathbf{x}_i) = (1+\mathbf{x}^T\mathbf{x}_i)^p$$

解 XOR 问题的指数 p 的最小值是多少？假定 p 为正整数。使用比最小值大的 p 值会出现什么结果？

6.10 图6-9表示三维模式 $\mathbf{x}$ 上运算的 XOR 函数，描述为

$$\mathrm{XOR}(x_1,x_2,x_3) = x_1 \oplus x_2 \oplus x_3$$

其中符号$\oplus$表示异或布尔函数运算符。设计一个多项式学习机，分离由这个运算符输出所表示的两类点。

6.11 在整个这一章中我们讨论利用支持向量机进行二分类。讨论支持向量机如何解决

M 类模式分类的问题($M>2$)。

非线性回归

6.12 在6.8节描述的利用支持向量机求解非线性回归问题的对偶问题，包括约束条件

349

$$\sum_{i=1}^{N}(\alpha_i - \alpha'_i) = 0$$

其中 α_i 和 α'_i 为 Lagrange 乘子。证明这个约束条件从对偏置 b 最小化 Lagrange 函数而得到，即对应于 $\varphi_0(\mathbf{x})=1$ 的权值向量 $\mathbf{w}$ 的第一个元素 w_0。

优点和局限

6.13 (a)就下列任务比较支持向量机和径向基函数(RBF)网络的优点和局限：(1)模式分类，(2)非线性回归。

(b)对于支持向量机和利用反向传播算法训练的多层感知器作同样比较。

计算机试验

6.14 图6-10表示两个类$\mathscr{C}_1$ 和$\mathscr{C}_2$ 的一组数据点。两个坐标轴 x_1 和 x_2 的范围都为 -1 到 $+1$。利用径向基函数核

$$K(\mathbf{x},\mathbf{t}) = \exp(-\|\mathbf{x}-\mathbf{t}\|^2)$$

对这个数据集构造最优超平面。

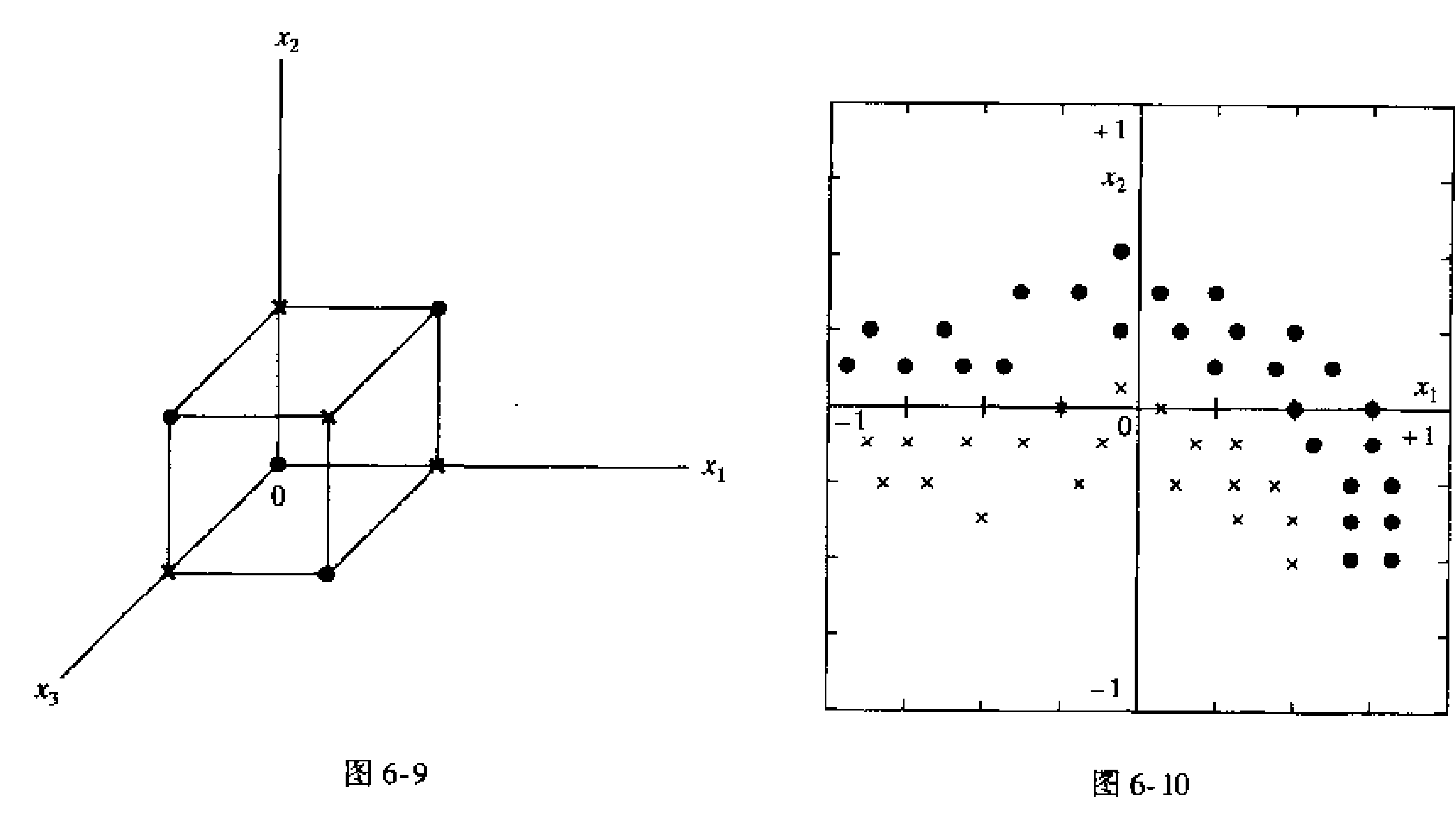


图6-9

图6-10

6.15 在6.6节描述的计算机实验是为了对两类部分重叠的 Gauss 分布进行分类。用于这个实验的正则化参数是 $C=0.1$。用于构造内积核函数的径向基函数的共有宽度为 $\sigma^2=4$。对于以下的两个正则化参数重复那一节中提到的计算机实验：(a) $C=0.05$，(b) $C=0.2$。根据6.6节报告的结果评论你的结果。

6.16 在用径向基函数网络求解非线性回归问题时，经常发现用多二次函数之类的非局部基函数比用 Gauss 函数之类的局部基函数导致更高精度解。对支持向量机可能猜想会出现类似的结果，因为利用(无界)多项式学习机可证明比(有界)径向基函数机器有更高精度。对模式分类问题用计算机实验探讨这个推测的正确性。

350

第7章 委员会机器

7.1 简介

在前面三章，我们描述三种不同的监督学习方法。在第4章，讨论由反方向传播算法训练的MLP，其设计依靠全局优化方式。在第5章，讨论RBF网络，其设计依靠局部优化方式。在第6章，讨论支持向量机，其设计利用VC维数理论。在本章我们将要提出另外一类解决监督学习任务的方法。这里使用的方法基于一个通常的工程原则：*分而治之*。

根据*分而治之的原则*(principle of divide and conquer)，一个复杂的计算任务被分解成一些简单的计算任务，然后再将这些任务的解重新组合起来。在监督学习中，我们将学习任务分配给一些专家以求得计算的简单化，这样就将输入空间划分成一组子空间。这些专家的组合就形成了*委员会机器*(committee machine)。从基本上说，它融合各专家所获得的知识使该机器能作出全局决策，可以设想这种决策优于任何一个专家单独作出的决策。这种"委员会机器"的思想可以追溯到Nilsson(1965)；那里考虑的网络结构是由一个基本的感知元层后面跟着在第二层的一个投票感知器组成的。

委员会机器是通用逼近器。它们可以被分成两大类：

1. *静态结构*。在这种委员会机器中，组合几个预报器(专家)响应的机制和输入信号无关，因此这种设计是"静态"的。这一类包括以下的方法：

- *总体平均*，其中将不同的预报器输出进行线形组合，产生整体输出。
- *推举*(boosting)方法，其中弱学习算法被转化为一个能达到任意高精确度的算法。

2. *动态结构*。在这第二种委员会机器中，将各单个专家输出组合成整体输出的机制直接和输入信号相关，因此名为"动态"。这里，我们将提到两种动态结构类：

- *混合专家*，所有专家的单独响应通过单个门网非线性地组合；
- *分层混合专家*，所有专家的单独响应通过多个门网层次式地非线性组合。

在混合专家中，分而治之的原则只被应用一次；而在分层混合专家中，分而治之的原则被应用多次，因而产生相应数量的层次。

混合专家网络和分层混合专家网络也可以被看作组合网络(modular network)的例子。组合性的(modularity)概念的正式定义是(Osherson et al.,1990)：

> *一个神经网络，只要它所进行的运算能分解成两个或者多个组件(module)(子系统)，各个组件有独立的输入变量，且相互之间没有通信，则称该神经网络是组合化的。各个组件的输出被一个整合单元调节，不允许向各个组件反馈信息。特别地，整合单元完成两项任务，(1)决定各个组件的输出怎样被整合，形成整个网络的最终输出，(2)决定哪些组件应学习哪些训练模式。*

这种组合性定义排除静态结构的委员会机器，因为它在输出端不存在具有决策作用的整合单元。

本章的组织

本章分为两个部分。第一部分为静态结构类，包括7.2节至7.5节。7.2节讨论总体平均的方法，其后7.3节是计算机实验。7.4节讨论推举技术，其后7.5节是计算机实验。

本章第二部分为动态结构类，包括7.6节至7.13节。具体地，7.6节讨论混合专家(ME)作为联想Gauss混合模型。7.7节讨论更一般的情况，即分层混合专家(HME)。这后一模型和标准决策树紧密相关。然后7.8节描述怎样对分层混合专家运用标准决策树求解HME的模型选择问题(即门网和专家网络的数目)。在7.9节我们定义*后验概率*，帮助我们对用于HME模型的学习方法建立公式。在7.10节通过对HME模型形成似然函数为解决参数估计问题奠定基础。7.11节给出学习策略的概览。随后在7.12节对EM算法进行详细讨论，在7.13节把这种算法应用于HME模型。

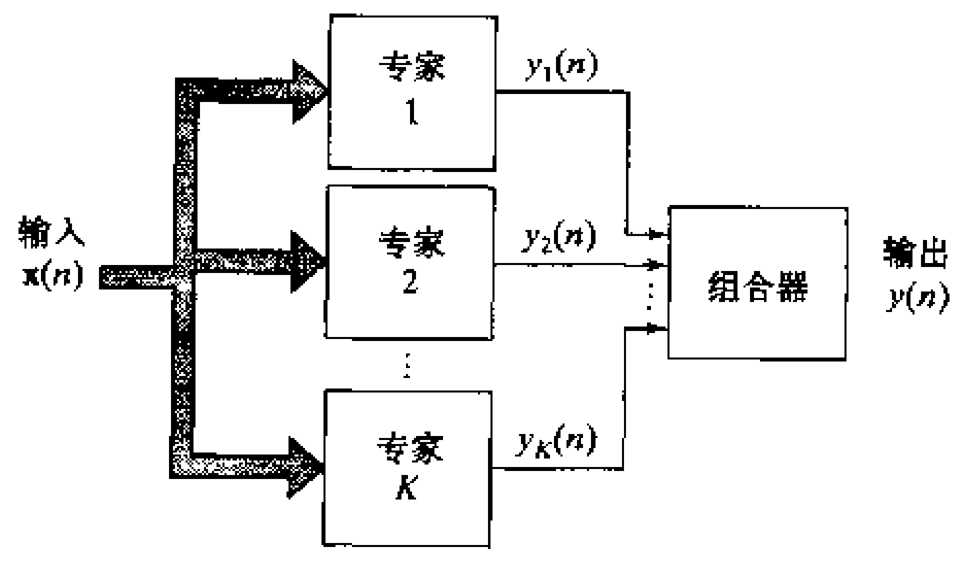
352 在7.14节以最后评论结束本章。

7.2 总体平均

图7-1显示了各种训练好的神经网络(即专家)，它们有一个共同的输入，然后将它们各自的输出整合成一个总的输出 y。为简化说明，这些专家的输出假定为标量值。这种技术被称作*总体平均方法*[1]。使用这种方法有双重动机：

- 假如图7-1中专家的整合用单个神经网络替代，我们将得到一个相对多的可调参数的网络。对这个一个大的网络进行训练的时间可能比并行训练一组专家的时间长。
- 当可调参数数目比训练数据集的基数(即集合的大小)大时，过拟合(overfitting)数据的风险也随之增大。

无论如何，在使用如图7-1描述的委员会机器时，我们期望分别训练的专家收敛到误差曲面的不同的局部极小，但整个系统性能通过将多个输出进行某种组合而得到提高。



353

图7-1 基于总体平均的委员会机器的框图

首先考虑对给定数据集合训练后的一个单独神经网络的情形。让 $\mathbf{x}$ 代表一个从来没有训练过的输入向量，让 d 代表一个相应期望输出(代表一个类的标号或者数值的响应)；$\mathbf{x}$ 和 d 分别代表随机向量 $\mathbf{X}$ 和随机变量 D 的实现。令 $F(\mathbf{x})$ 代表网络所实现的输入-输出函数。根据第2章提到的"偏置/方差"困境的知识，我们可以把 $F(\mathbf{x})$ 和条件期望 $E[D \mid \mathbf{X}=\mathbf{x}]$ 的均方误差分解成偏置和方差分量如下：

$$E_{\mathcal{D}}[(F(\mathbf{x}) - E[D \mid \mathbf{X} = \mathbf{x}])^2] = B_{\mathcal{D}}(F(\mathbf{x})) + V_{\mathcal{D}}(F(\mathbf{x})) \tag{7.1}$$

其中 $B_{\mathscr{D}}(F(\mathbf{x}))$是偏置的平方：

$$B_{\mathscr{D}}(F(\mathbf{x})) = (E_{\mathscr{D}}[F(\mathbf{x})] - E[D \mid \mathbf{X} = \mathbf{x}])^2 \tag{7.2}$$

而 $V_{\mathscr{D}}(F(\mathbf{x}))$是方差：

$$V_{\mathscr{D}}(F(\mathbf{x})) = E_{\mathscr{D}}[(F(\mathbf{x}) - E_{\mathscr{D}}[F(\mathbf{x})])^2] \tag{7.3}$$

期望 $E_{\mathscr{D}}$对空间$\mathscr{D}$取期望，$\mathscr{D}$被定义为包括所有的训练集(即输入和目标输出)的分布和所有的初始条件分布的空间。

有多种单独训练图 7-1 中专家的方法，也有多种合并其输出的方法。在这里的讨论中，我们考虑所有的专家网络有相同的构形(结构)的情况，但它们是从不同的初始条件开始训练的。在图 7-1 所示的委员会机器输出的组合器中，仅用简单的总体平均器(ensemble averager)[2]。令$\mathscr{I}$代表所有初始条件的空间。令 $F_I(\mathbf{x})$代表图 7-1 中专家网络的输入－输出函数在一系列"有代表性"的初始条件下的平均。和式(7.1)类似，可以写出

$$E_{\mathscr{I}}[(F_I(\mathbf{X}) - E[D \mid \mathbf{X} = \mathbf{x}])^2] = B_{\mathscr{I}}(F(\mathbf{x})) + V_{\mathscr{I}}(F(\mathbf{x})) \tag{7.4}$$

其中 $B_{\mathscr{I}}(F(\mathbf{x}))$是定义在空间$\mathscr{I}$的偏置的平方：

$$B_{\mathscr{I}}(F(\mathbf{x})) = (E_{\mathscr{I}}[F_I(\mathbf{x})] - E[D \mid \mathbf{X} = \mathbf{x}])^2 \tag{7.5}$$

而 $V_{\mathscr{I}}(F(\mathbf{x}))$是方差

$$V_{\mathscr{I}}(F(\mathbf{x})) = E_{\mathscr{I}}[(F_I(\mathbf{x}) - E_{\mathscr{I}}[F(\mathbf{x})])^2] \tag{7.6}$$

期望 $E_{\mathscr{I}}$是对空间$\mathscr{I}$取期望。

从空间$\mathscr{D}$的定义，我们可以将它看作初始条件所在的空间$\mathscr{I}$和表示为$\mathscr{D}'$的剩余空间(remnant space)的乘积。因此，再次通过与式(7.1)相似性，可以写出

$$E_{\mathscr{D}'}[(F_I(\mathbf{x}) - E[D \mid \mathbf{X} = \mathbf{x}])^2] = B_{\mathscr{D}'}(E_I(\mathbf{x})) + V_{\mathscr{D}'}(F_I(\mathbf{x})) \tag{7.7}$$

其中 $B_{\mathscr{D}'}(F_I(\mathbf{x}))$是定义在剩余空间$\mathscr{D}'$上的偏置平方：

$$B_{\mathscr{D}'}(F_I(\mathbf{x})) = (E_{\mathscr{D}'}[F_I(\mathbf{x})] - E[D \mid \mathbf{X} = \mathbf{x}])^2 \tag{7.8}$$

而 $V_{\mathscr{D}'}(F_I(\mathbf{x}))$是相应的方差：

$$V_{\mathscr{D}'}(F_I(\mathbf{x})) = E_{\mathscr{D}'}[(F_I(\mathbf{x}) - E_{\mathscr{D}'}[F_I(\mathbf{x})])^2] \tag{7.9}$$

从空间$\mathscr{D}$，$\mathscr{I}$和$\mathscr{D}'$的定义容易看出

$$E_{\mathscr{D}'}[F_I(\mathbf{x})] = E_{\mathscr{D}}[F(\mathbf{x})] \tag{7.10}$$

因此随之可将式(7.8)重写为下列等价形式：

$$B_{\mathscr{D}'}(F_I(\mathbf{x})) = (E_{\mathscr{D}}[F(\mathbf{x})] - E[D \mid \mathbf{X} = \mathbf{x}])^2 = B_{\mathscr{D}}(F(\mathbf{x})) \tag{7.11}$$

接下来考虑式(7.9)中的方差 $V_{\mathscr{D}'}(F_I(\mathbf{x}))$。由于随机变量的方差等于随机变量的均方值减去它的偏置的平方，可以等价写为

$$V_{\mathscr{D}'}(F_I(\mathbf{x})) = E_{\mathscr{D}'}[(F_I(\mathbf{x}))^2] - (E_{\mathscr{D}'}[F_I(\mathbf{x})])^2 = E_{\mathscr{D}'}[(F_I(\mathbf{x}))^2] - (E_{\mathscr{D}}[F(\mathbf{x})])^2 \tag{7.12}$$

354

其中在第二个等式利用了式(7.10)，类似地我们可以以等价的形式重新定义式(7.3)：

$$V_{\mathscr{D}}(F_I(\mathbf{x})) = E_{\mathscr{D}}[(F(\mathbf{x}))^2] - (E_{\mathscr{D}}[F(\mathbf{x})])^2 \tag{7.13}$$

注意函数 $F(\mathbf{x})$在整个空间$\mathscr{D}$上的均方值一定大于或者等于整体均方函数 $F_I(\mathbf{x})$在剩余空间$\mathscr{D}'$

上的均方值，即

$$E_{\mathscr{D}}[F(\mathbf{x})^2] \geq E_{\mathscr{D}}[(F_I(\mathbf{x}))^2]$$

根据这个不等式，比较式(7.13)和(7.12)，立即可以得到

$$V_{\mathscr{D}'}(F_I(\mathbf{x})) \leq V_{\mathscr{D}}(F(\mathbf{x})) \tag{7.14}$$

根据式(7.11)和(7.14)，我们可以作出下面的两个结论：

1. 属于如图 7-1 的委员会机器的总体平均函数 $F_I(\mathbf{x})$的偏置正好和属于一个单个神经网络的函数 $F(\mathbf{x})$的偏置相同。

2. 总体平均函数 $F_I(\mathbf{x})$的方差小于函数 $F(\mathbf{x})$的方差。

这些理论发现指出一个用于减少委员会机器产生的总误差的训练策略是由不同的初始条件得到的(Naftaly et al.,1997)。机器的专家成员被故意过度训练，使用它的理由是基于下面的基础。只要考虑单个专家，偏置的减少就是以方差为代价的。但是，此后通过对初始条件总体平均专家，方差减少了而偏置保留不变。

7.3 计算机实验 I

在关于总体平均方法的计算机实验中，我们重新回到前面三章考虑的模式分类问题。问题属于两个有重叠的二维 Gauss 分布的分类问题。这两个分布有着不同的均值向量和不同的方差。分布 1(类$\mathscr{C}_1$)的统计特性为

$$\boldsymbol{\mu}_1 = [0,0]^T, \sigma_1^2 = 1$$

[355] 分布 2(类$\mathscr{C}_2$)的统计特性为

$$\boldsymbol{\mu}_2 = [2,0]^T, \sigma_2^2 = 4$$

两个分布的散列图在图 4-13 给出。

这两类被假定为等概率的。错误分类的代价假定相同，正确分类的代价假定为 0。在此基础上，(最优)贝叶斯分类器有 $p_c = 81.51\%$ 的正确分类率。这个计算的细节已经在第 4 章给出。

在第 4 章描述的计算机实验中，应用有两个隐藏神经元的多层感知器和使用反向传播算法训练，我们能得到将近 80% 的正确分类率。在这个实验中，我们将学习一个如下组成的委员会机器：

- 10 个专家。
- 每个专家由一个具有两个隐藏单元的多层感知器组成。

所有的专家都应用反向传播算法进行单独训练。算法中使用的参数是学习率参数 $\eta = 0.1$，动量常数 $\alpha = 0.5$。

训练样本的大小是 500 个模式。所有的专家在同一个数据集上训练，只不过它们的初始条件不同。特别地，初始权值和阈值是随机地从区间[-1,1]按均匀分布随机挑选的。

表 7-1 汇总 10 个专家通过使用测试集的 500 个模式训练后的分类性能。仅靠简单地提取表 7-1 中 10 个结果后算术平均而得到的正确分类率为 $p_{c,av} = 79.37\%$。另一方面，应用总体平均方法，即简单地将各个专家的输出相加后计算正确分类率，我们得到结果：$p_{c,ens} = 80.27\%$。这个结果比 $p_{c,av}$ 提高了 0.9 个百分点。这种改进对所有的实验来说都是存在的。分类结果是应用 32 000 个测试模式计算出来的。

总结这个实验的结果，我们可以说：通过过度训练单个多层感知器(专家)，将它们各自的数值输出相加产生委员会机器的总输出，然后作出决策，由此提高分类性能。

表 7-1 在委员会机器中使用的单个专家的分类性能

专家	正确分类的百分数
Net1	80.65
Net2	76.91
Net3	80.06
Net4	80.47
Net5	80.44
Net6	76.89
Net7	80.55
Net8	80.47
Net9	76.91
Net10	80.38

356

7.4 推举

如同在介绍中提到的那样，推举是属于静态结构的委员会机器的另一种方法。推举和总体平均有很大的不同。在基于总体平均的一个委员会机器中，所有的专家在一个数据集上训练，在训练的过程中，它们是由于初始条件不同而导致不同的。与此相反，推举机器中的专家各自的训练集是完全不同的分布；它是能被用来提高任何学习算法性能的一个通用方法。

推举[3](boosting)能用三种基本不同的方法实现：

1. *通过过滤推举*。这种方法涉及到用一个弱学习算法的不同版本过滤训练样本。它假定有大量(理论上无穷)样本可用，这些样本在训练过程中有些被抛弃，有些被保留。这个方法比另外两种方法的一个优越之处在于它具有较小的存储需求。

2. *通过子抽样推举*。第二种方法用到一个固定大小的训练样本集合。训练过程中这些样本根据一个给定概率分布“重新抽样”。根据固定的训练样本计算误差。

3. *通过重新加权推举*。第三种方法也用到一个固定大小的训练样本集合，但它假定弱学习算法能接收“加权”后的样本。根据加权后的样本计算误差。

在这一节将描述两种不同的推举算法。其中之一归功于 Schapire(1990)，属于方法 1；另外的一种称为自举(AdaBoost)，归功于 Freund and Schapire(1996a,1996b)，属于方法 2。

通过过滤推举

在 Schapire(1990)描述的推举，其基本思想植根于一个*与分布无关的或可能近似正确的*(probably approximately correct, PAC)*学习模型*。通过在第二章讨论过的 PAC 学习，我们知道一个概念(concept)只是某范例(instance)域内的一个布尔函数，该范例域包括我们感兴趣的所有对象(object)的编码。在 PAC 学习中，一个学习机器通过随机选择概念的样本的基础上，去确认一个未知的二值概念。更进一步地说，学习机器的目标是找到一个错误率最多为 ϵ 的假说或者预测规则，ϵ 为任意小的正数，并且它对于所有输入分布都是一致成立的。基于此，PAC 学习模型又称为*强学习模型*(strong learning model)。因为样本的随机性质，那么极有可能由于一些高度不具有代表性的样本存在而不能学到有关未知概念的任

357

何东西。因此我们要求学习模型只在以概率 $1-\delta$ 找到未知概念的良好近似后继续，这里 δ 是一个小的正数。

在 PAC 学习模型中，有一个变形称为*弱学习模型*(weak learning model)。它对于学习未知概念的要求大大地放松了。现在学习机器被要求以稍微小于 1/2 的误差率去发现一个假说。当一个假说对于每一个例子以完全随机的方式去猜想一个二值的标号时，它错误和正确的概率是相同的。也就是说，它得到一个恰好 1/2 的误差率。从而，随之而来的弱学习模型实际表现只比随机猜想仅略好一点。弱可学习的概念是 Kearns and Valiant(1989)引入的，他们提出了*假说推举问题*，它在下面的问题中体现出来：

弱学习和强学习二者概念等价吗？

换句话说，任何是弱可学习的概念类，是否也是强可学习的？或许是惊奇的，这个问题由 Schapire(1990)肯定地回答了。其证明是构造性的。特别地，一个直接将弱学习模型转化成强学习模型的算法被设计出来。它的取得是通过改变样本的分布使得由一个弱学习模型建立一个强学习模型。

在基于过滤的推举中，委员会机器由三个专家或子假说组成。用于训练它们的算法称为*推举算法*(boosting algorithm)。这三个专家可随意标为"第一"、"第二"和"第三"。这三个专家各自训练如下：

1. 第一个专家在 N_1 个样本上训练。

2. 被训练过的第一个专家通过下面的方式过滤另外一个样本集：

- 抛一枚硬币；这实际是模拟一个随机猜测。
- 假如结果是*正面*，则新模式通过第一个专家，并抛弃被正确分类的模式，直到遇到一个被错误分类的模式为止。这个错误分类模式被加入到第二个专家的训练集中。
- 假如结果是*反面*，所做的恰好相反。特别地，将新模式通过第一个专家，抛弃不能被正确分类的模式，直到遇到一个能被正确分类的模式为止。正确分类的模式被加入到第二个专家的训练集中。
- 继续这个过程，直到 N_1 个样本被第一个专家过滤，这个过滤后的样本组成第二个专家的训练集。

依据抛硬币过程，可以确保假如第一个专家在第二个样本集上测试，它将有 1/2 的误差率。换句话说，用来训练第二个专家的第二个含有 N_1 样本的集合和第一个用来训练第一个专家的 N_1 样本的集合具有完全不同的分布。用这种方法，第二个专家被强制学习和第一个专家的分布完全不同的分布。 [358]

3. 一旦第二个专家通过正常方式训练完毕，供第三个专家使用的第三个训练集将通过如下的方式产生：

- 将一个新的模式通过专家 1 和专家 2。假如这两个专家的决策一致，则抛弃该模式，否则该模式被加入到第三个专家的训练集中。
- 继续这个过程，直到 N_1 个样本被第一个专家和第二个专家所共同过滤。这个被过滤得到的样本集组成第三个专家的训练集。

这个三步过滤过程如图 7-2 所示。

令 N_2 代表一个样本集的数目，该样本集必须被第一个专家过滤以便得到供第二个专家

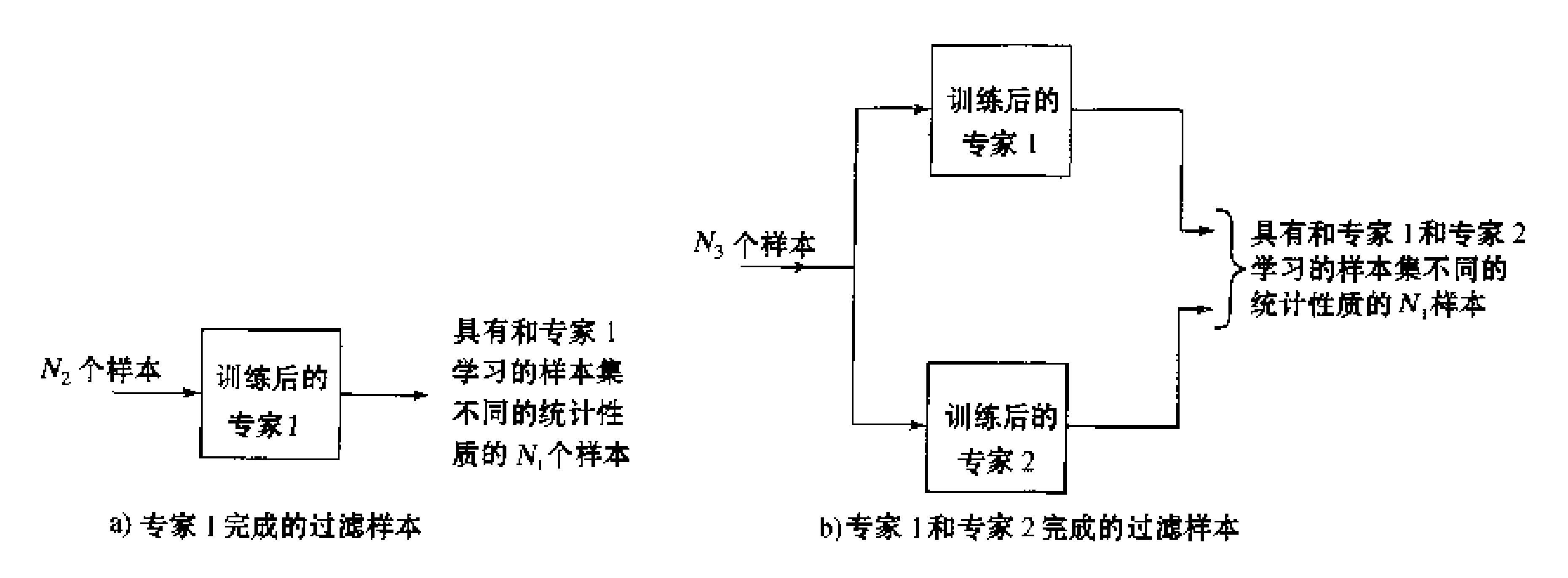


图 7-2 通过过滤的推举示意图

使用的 N_1 个样本训练集。注意 N_1 是固定的，N_2 取决于第一个专家的泛化误差率。令 N_3 代表一个样本集的数目，该样本集必须被第一个和第二个专家所共同过滤而得到供第三个专家训练用的 N_1 样本集。因为 N_1 个样本需要用来训练第一个专家，总共需要用来训练委员会机器的训练集的大小为 $N_4 = N_1 + N_2 + N_3$。但计算的代价是基于 $3N_1$ 个样本，因为 N_1 正好是用来分别训练三个专家的样本的数目。委员会机器需要一个很大的样本集供其操作，但仅仅是该样本集的一个子集被用来实施真正的训练，从这一点上来说，我们可以说这里描述的推举算法确实是"聪明"的。

另一点值得注意的是，通过第一个专家网络的过滤操作和通过第一和第二个专家联合的过滤操作，使得第二个和第三个专家网络能分别集中学习分布中"难以学习"的部分。 359

在最早由 Schapire(1990)提出的推举算法的理论推导中，用简单表决来评估委员会机器对于未学习过的测试模式的性能。特别地，一个测试模式被提交给委员会机器，假如第一个和第二个专家各自的决策相一致，则使用这个类的标号。否则，使用第三个专家发现的类的标号。但是，由 Drucker et al.(1993,1994)给出的实验工作确定将三个专家各自的输出相加将会产生比表决更好的性能。比如说，在光学字符识别(OCR)问题中，相加运算仅仅只对三个专家"数字 0"的输出相加，另外的 9 个数字的输出也是同样的。

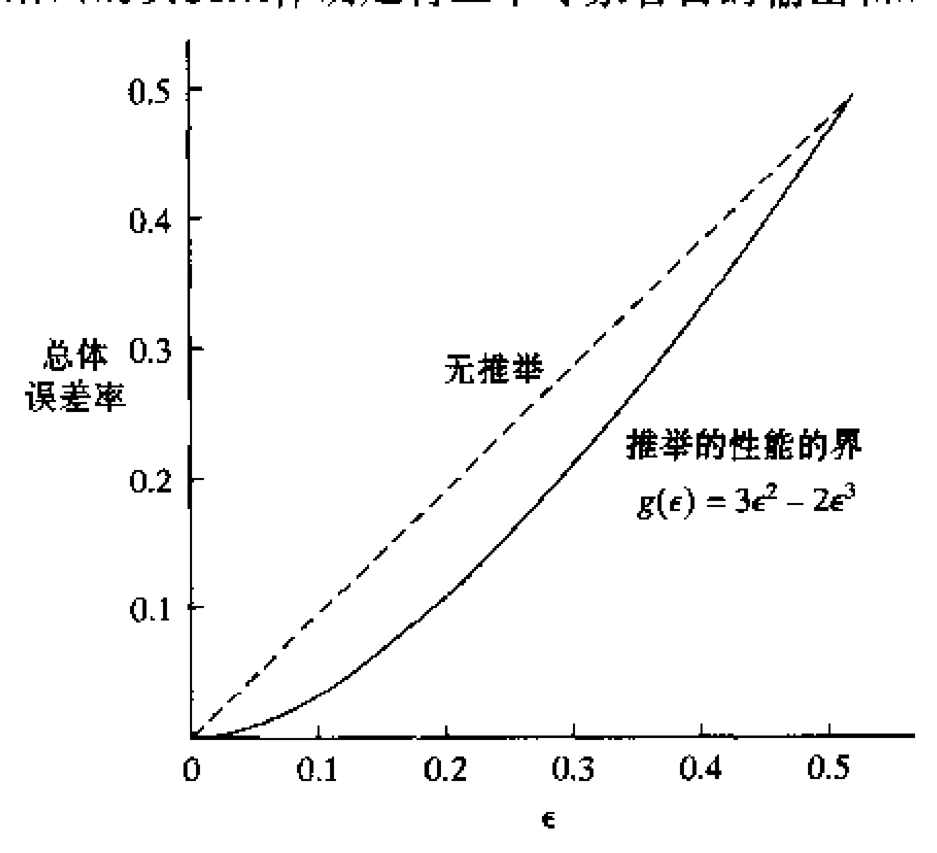


图 7-3 式(7.15)通过滤波推举的图形

假如三个专家(即子假说)在它们各自训练的分布上误差率为 $\varepsilon < 1/2$；也就是说，它们三个都是弱学习模型。在 Schapire(1990)中证明委员会机器的总误差率以

$$g(\varepsilon) = 3\varepsilon^2 - 2\varepsilon^3 \qquad (7.15)$$

为界。界 $g(\varepsilon)$ 相对 ε 的图形如图 7-3 所示。从该图中，我们可以看出界比原始误差率 ε 小得多。通过递归运用

推举算法，这个误差率能变得任意小。一个弱学习模型，其性能仅仅比随机猜想好一点，被转换成一个强学习模型。在这个意义上我们可以说强学习模型和弱学习模型确实是等价的。

自举

通过过滤推举的一个实际的局限在于它经常需要大量的训练样本。这种局限能通过利用另外一种称为自举(AdaBoost)的推举算法克服(Freund and Schapire,1996a,1996b)，它属于重新采样的推举。自举的采样框架是集中式学习的自然框架；最重要的是，它允许训练数据重用。

360

和通过过滤算法推举一样，自举方法也用于弱学习模型。这个新方法的目的是找到一个对给定的带标号样本的分布$\mathscr{D}$具有低误差率的最终映射函数或假说。它在两个方面和其他的推举不同。

- 自举自适应调节由弱学习模型返回的弱假设误差，这就是算法名称的由来。
- 自举性能的界只取决于弱学习模型对学习过程中实际产生的那些分布的性能。

自举操作如下。对于迭代 n，推举算法提供在训练样本$\mathscr{T}$上分布为$\mathscr{D}_n$的弱学习模型。作为响应该弱学习模型计算一个假说$\mathscr{F}_n$：$\mathbf{X}\rightarrow Y$，它能正确地分类训练样本的一部分。误差通过分布$\mathscr{D}_n$来度量。这个过程持续 T 次迭代，最后推举机器将这些假说$\mathscr{F}_1,\mathscr{F}_2,\cdots,\mathscr{F}_T$合并成一个最终的假说$\mathscr{F}_{\text{fin}}$。

为了计算(1)对迭代 n 上的分布$\mathscr{D}_n$，和(2)最终的假说$\mathscr{F}_{\text{fin}}$，使用表7-2小结的简单过程。初始分布$\mathscr{D}_1$是训练样本$\mathscr{T}$上的均匀分布，表示为

$$\mathscr{D}_1(i)=\frac{1}{n}\qquad \text{对于所有的 } i$$

给定算法在迭代 n 的分布$\mathscr{D}_n$和弱假说$\mathscr{F}_n$，如果弱假说$\mathscr{F}_n$能正确分类输入向量 $\mathbf{x}_i$，则下一个分布$\mathscr{D}_{n+1}$中对例子 i 的权重乘以一个数 $\beta_n\in[0,1]$；否则，权值不变。然后通过将权值除以归一化常数 Z_n 而重新归一化。实际上，训练集$\mathscr{T}$总被许多先前的弱假说正确地分类的"容易"的样本赋予较低权值，而被经常错误分类的"难"的样本被赋予了较高的权值。因此自举算法将更多的权值集中到看起来最难分类的样本上。

至于最终假说$\mathscr{F}_{\text{fin}}$，它是根据弱假说$\mathscr{F}_1,\mathscr{F}_2,\cdots,\mathscr{F}_T$加权表决的方式(即加权线性阈值)计算的。也就是说，对于一个给定的输入向量 $\mathbf{x}$，最终假说$\mathscr{F}_{\text{fin}}$输出的标号 d 使得预测该输出标号的弱假说的加权求和为最大。假说$\mathscr{F}_n$的权值定义为 $\log(1/\beta_n)$，结果是较大的权值被赋予较低误差率的假说。

361

自举的一个重要理论性质如下面定理所述(Freund and Schapire,1996a)：

假如一个弱学习模型，当被自举调用时，产生误差为$\varepsilon_1,\varepsilon_2,\cdots,\varepsilon_T$的假说，其中自举算法在迭代 n 时误差 ε_n 定义为

$$\varepsilon_n=\sum_{i:\mathscr{F}_n(\mathbf{x}_i)\neq d_i}\mathscr{D}_n(i)$$

假设 $\varepsilon_n\leqslant 1/2$，且令 $\gamma_n=1/2-\varepsilon_n$。那么最终假说误差的如下上界成立：

$$\frac{1}{N}\left|\{i:\mathscr{F}_{\text{fin}}(\mathbf{x}_i)\neq d_i\}\right|\leqslant\prod_{n=1}^{T}\sqrt{1-4\gamma_n^2}\leqslant\exp\left(-2\sum_{n=1}^{T}\gamma_n^2\right)\tag{7.16}$$

这个定理证明通过弱学习模型构造的弱假说只要恒有着比 1/2 稍微好一点的误差，则最终假说$\mathcal{F}_{fin}$的训练误差呈指数级下降趋于 0。但是，这并不意味着测试数据上的泛化误差必定小。在 Freund and Schapire(1996a)中给出的实验表明两点。第一，训练误差的理论界经常是很弱的；第二，泛化误差倾向于比该理论暗示的误差好得多。

表 7-2 给出用于一个二值分类问题的自举的小结。

表 7-2 自举方法小结

输入：训练样本$\{(\mathbf{x}_i, d_i)\}_{i=1}^N$

N 个标记样本的分布$\mathcal{D}$

弱学习模型

整数 T 指定算法的迭代次数

初始化：对于所有的 i，置$\mathcal{D}_1(i) = 1/N$

计算：对于$n = 1, 2, \cdots, T$，进行下面的过程：

1. 调用弱学习模型，对它提供分布$\mathcal{D}_n$
2. 返回假说$\mathcal{F}_n$：$\mathbf{X} \rightarrow Y$
3. 计算假说$\mathcal{F}_n$ 的误差

$$\varepsilon_n = \sum_{i:\mathcal{F}_n(\mathbf{x}_i) \neq d_i} \mathcal{D}_n(i)$$

4. 设置 $\beta_n = \varepsilon_n/(1-\varepsilon_n)$
5. 更新分布$\mathcal{D}_n$：

$$\mathcal{D}_{n+1}(i) = \frac{\mathcal{D}_n(i)}{Z_n} \times \begin{cases} \beta_n & 若\mathcal{F}_n(\mathbf{x}_i) = d_i \\ 1 & 否则 \end{cases}$$

其中 Z_n 是归一化常数(选择它使得$\mathcal{D}_{n+1}(i)$是一概率分布)。

输出：最终的假说是

$$\mathcal{F}_n(\mathbf{x}) = \arg\max_{d \in \mathcal{D}_n} \sum_{\mathcal{F}_n(\mathbf{x}) = d} \log \frac{1}{\beta_n}$$

当可能的类别(标号)$M > 2$时，推举问题变得更复杂，因为随机猜想给出正确标号的概率是 $1/M$，比 1/2 要小。在这种情况下为了推举能使用任何比随机猜想好一点点的假说，我们就需要改变算法和“弱学习”算法是什么的定义。使用改变的方法在 Freund and Schapire (1997)以及 Schapire(1997)中描述。

误差特性

在 Breiman (1996b)中报告的自举方法的实验表明，当训练误差和测试误差作为推举迭代次数的函数时，我们经常发现当训练误差实质上减小为 0 后，测试误差继续下降。这种现象如图 7-4 所显示。对于通过过滤的推举，Drucker et al.(1994)更早报导过类似的结果。 362

根据我们所知道的单个神经网络的一般特性来说，图 7-4 所显示的现象是令人惊讶的。回想第 4 章，在用反向传播算法训练多层感知器时，测试(确认)数据的误差先减少，到达一个最小值，然后由于过拟合而上升；可以参看图 4-20。图 7-4 所示的情况是很不同的，随着网络通过不断的训练变得越来越复杂，推广误差持续下降。这种现象似乎和“Occam剃刀原

理"相冲突，该原理表明，一个学习机器应尽可能地简单，以便于达到一个好的泛化性能。

在 Schapire et al.(1997)中，给出对这个用于自举的现象的一种解释。那里提出的分析的重要思想是，当计算一个推举机器产生的泛化误差时，不仅仅要考虑训练误差，还要考虑分类的置信度(confidence)。提出的分析揭示推举和支持向量机之间的关系；支持向量机已经在前一章考虑。特别地，比如分类边界定义为赋予属于那个样本的正确标号的权值和赋予任一不正确标号的最大权值的差。从这个定义，容易看出边界是区域[-1, 1]内的一个数，并且如果一个样本能被正确分类的充分必要条件是它的边界是正的。因此 Schapire 等人证明在图 7-4 中观察到的现象确实和产生表决分类误差的训练样本的边界分布有关。需要再次强调的是 Schapire et al.(1997)给出的边界分析只是针对自举的和不适用于其他推举的算法。

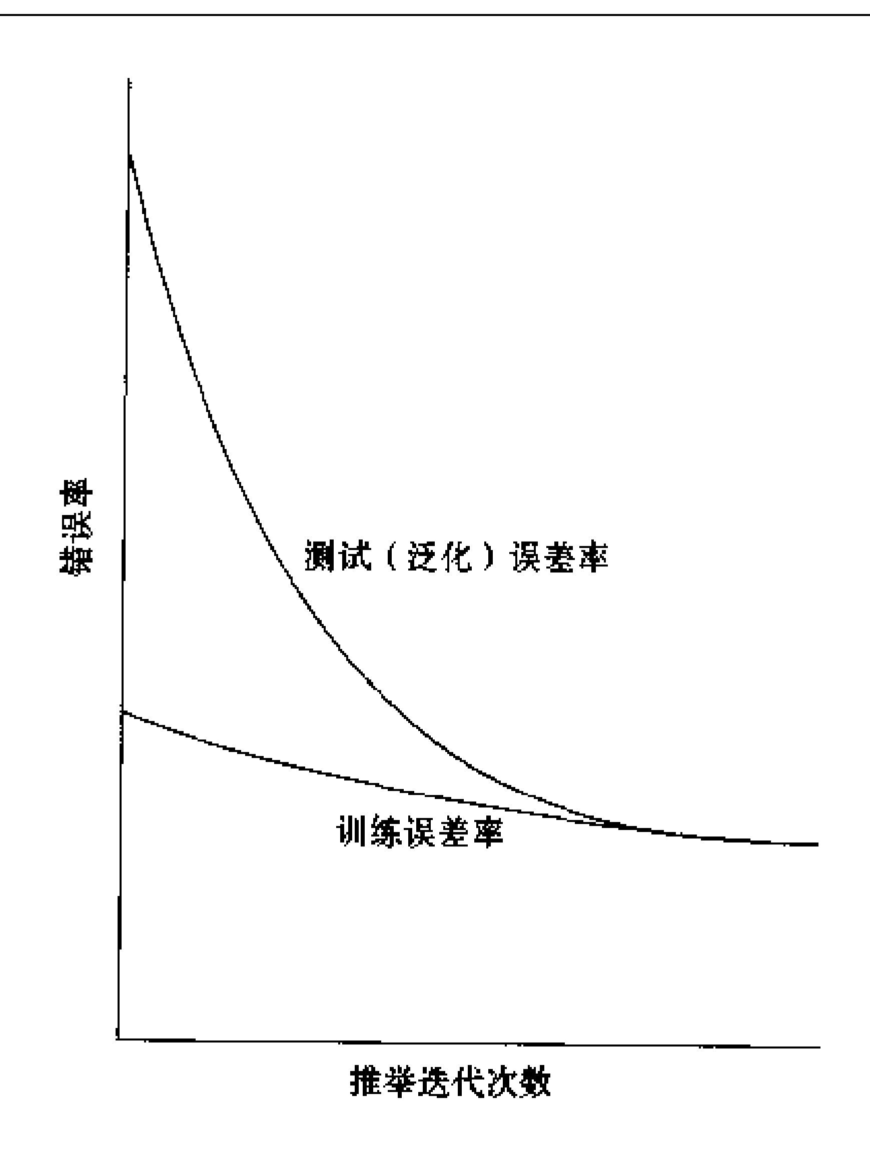


图 7-4 自举算法的概念化误差特性

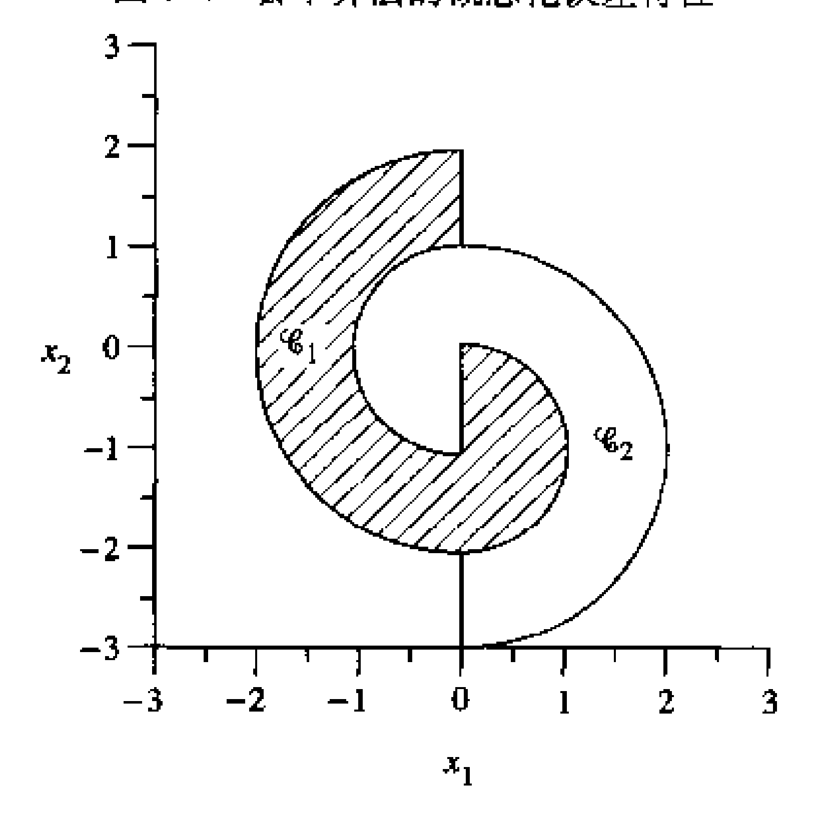


图 7-5 用于推举试验的模式构形

7.5 计算机实验 II

在这个实验中，我们将运用通过过滤的推举算法解决一个相当难的模式分类任务。分类问题是二维的包含非凸的决策区域，如图 7-5 所示。一类模式由位于标号为 $\mathscr{C}_1$ 的区域内的数据点组成，另外一类模式由位于标号为 $\mathscr{C}_2$ 的区域内的数据点组成。要求设计一个委员会机器，用于决定一个测试模式属于类 $\mathscr{C}_1$ 或类 $\mathscr{C}_2$。

用于解决这个问题的委员会机器由三个专家组成。每一个专家包含由两个输入节点、五个隐藏神经元和两个输出神经元组成的 2-5-2 多层感知器。应用反向传播算法完成训练。图 7-6 显示用来训练三个专家的数据散布图。图 7-6a 所示数据用于训练专家 1。图 7-6b 所示数据是经过在专家 1 完成训练后过滤得到的；这些数据用于训练专家 2。图 7-6c 所示的数据是由专家 1 和专家 2 所共同过滤后用来训练专家 3 的。对于每一个专家来说，训练样本的大小都是 $N_1 = 1000$ 个模式。仔细检查这三个图我们可以观察到：

- 图 7-6a 中用于专家 1 的训练数据是均匀分布的。

- 图 7-6b 中用专家 2 的训练数据，显示在标有 A 和 B 的区域内数据点的集中，这对于专家 1 分类来说似乎是很困难的。在这两个区域内的数据点的数目等于被正确分类的点的数目。 364
- 图 7-6c 中用于专家 3 的训练数据，显示数据点更加集中，看起来对于专家 1 和专家 2 分类来说都是困难的。

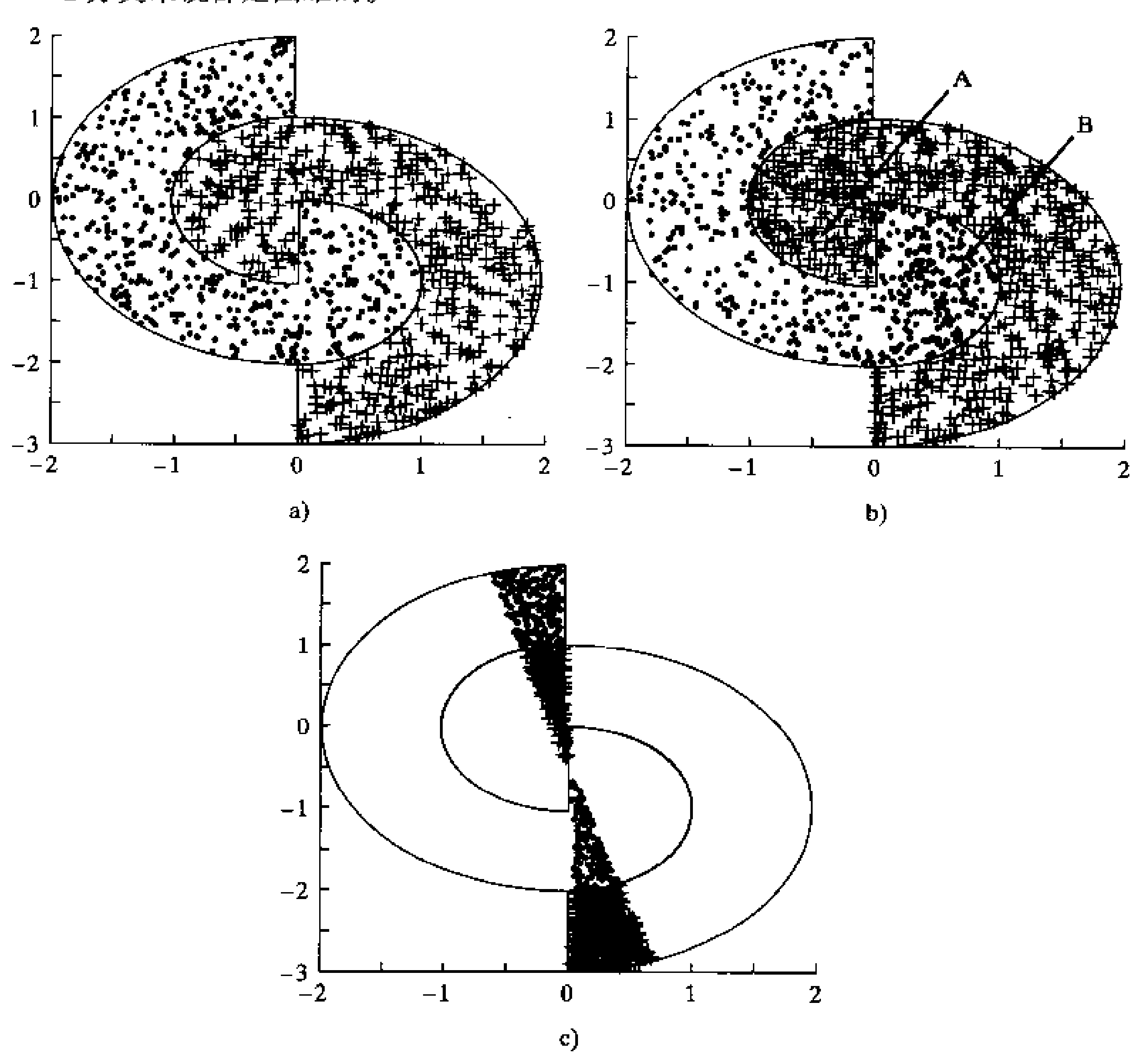


图 7-6 推举的计算机试验中用于专家训练的样本散布图

a) 专家 1 b) 专家 2 c) 专家 3

图 7-7a、7-7b、7-7c 显示专家 1、专家 2 和专家 3 各自形成的决策边界。7-7d 显示通过将三个专家输出进行简单相加而形成的总体决策边界。注意，属于专家 1 和专家 2 的决策区域 7-7a 和 7-7b 之间的差异定义用来训练专家 3 的图 7-7c 的训练数据点的分布。

三个专家对于测试数据正确分类的概率是：

专家 1：75.15%，专家 2：71.44%，专家 3：68.90%

整个委员会机器的正确分类概率是 91.79%，它是用 32 000 个模式的测试数据计算得到的。图 7-7d 所示的三个专家的推举算法建立的总体决策边界，进一步证明它的良好分类性能。 365

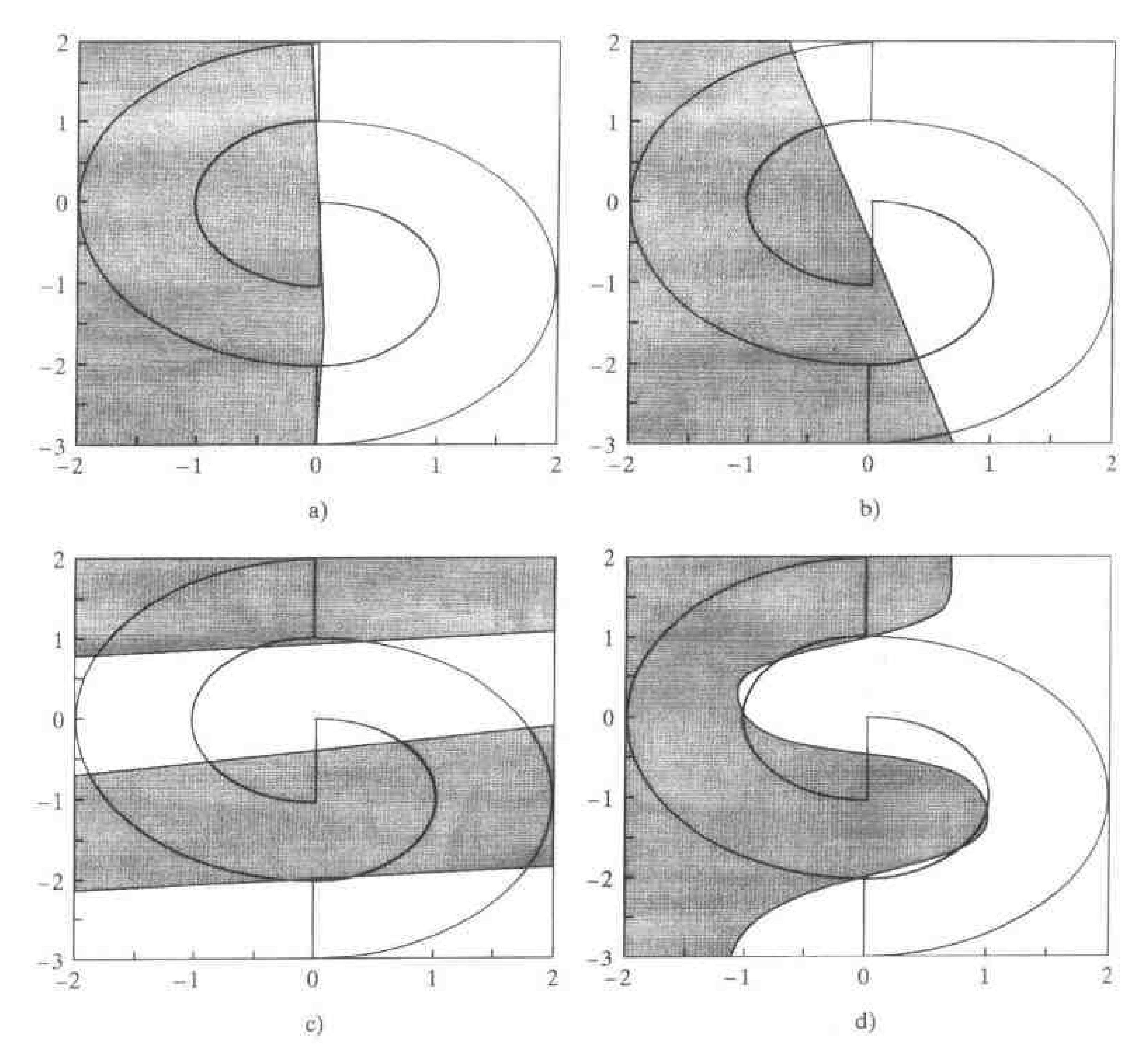


图 7-7 在推举试验中不同专家形成的决策边界
a)专家 1 b)专家 2 c)专家 3 d)整个委员会机器

7.6 联想 Gauss 混合模型

从本节开始的本章第二部分我们研究第二类委员会机器，即动态结构。用在这里的术语“动态”是指专家的知识整合是在输入信号的参与作用下完成的。

为了开始我们的讨论，考虑一个组合网络，在其中学习过程是通过将学习的自组织和监督形式以无缝方式融合在一起处理的。各个专家从技术上进行监督学习，把它们各自的输出整合以模拟期望响应。但是各个专家也进行自组织学习；即它们自组织地发现一个好的输入空间的分割，以便于每个专家能很好地模拟它自己的子空间，而且作为一个完整的组它们能很好地模拟输入空间。

366

在刚才描述的学习方案中，有一点和前面三章讨论的学习方案不同，那就是假设用一个特殊的模型产生训练数据。

概率产生模型

为了确定概念，考虑一个回归问题，其中一个回归量 $\mathbf{x}$ 产生用随机变量 D 表示的响应；

这个随机变量的一个实例用 d 来表示。为了简化表达，并不失一般性，我们采用一个标量形式的回归。我们假设响应 d 的产生遵循下列的概率模型(Jordan and Jacobs, 1995)：

1. 输入向量 $\mathbf{x}$ 随机地从某一先验分布中选取。

2. 给定 $\mathbf{x}$ 和某个参数向量 $\mathbf{a}^{(0)}$，根据条件概率 $P(k|\mathbf{x},\mathbf{a}^{(0)})$ 选定某个特定的规则，比如说第 k 个规则。

3. 对于规则 k，$k=1,2,\cdots,K$，模型响应 d 和 $\mathbf{x}$ 是线形关系，并且有一个附加的误差 ε_k，ε_k 模拟成 Gauss 随机分布的随机变量，其均值为 0，方差为单位值 1：

$$E[\varepsilon_k]=0 \qquad \text{对于所有的 } k \tag{7.17}$$

和

$$\text{var}[\varepsilon_k]=1 \qquad \text{对于所有的 } k \tag{7.18}$$

第 3 点作出单位方差的假设只是为了讲解的简洁性。一般地，每一个专家都有能从训练数据中学习的一个不同的输出方差。

给定 $\mathbf{x}$ 和某个参数向量 $\mathbf{w}_k^{(0)}$，$k=1,2,\cdots,K$，D 的概率产生取决于条件概率 $P(D=d|\mathbf{x},\mathbf{w}_k^{(0)})$。我们并不要求刚才描述的概率产生模型必须是对物理现实的一个直接的对应。相反，我们仅仅要求在那里包含的概率决策能表示一个抽象模型，它以递增的精确度确定一个非线性流形上响应 d 的条件均值，这个非线性流形建立输入向量和均值输出的关系(Jordan, 1994)。

根据这个模型，对应于标号 k 的 K 个选择，响应 D 能产生 K 个不同的方法。因此，在给定输入向量 $\mathbf{x}$ 的情况下，产生响应 $D=d$ 的条件概率等于

$$P(D=d|\mathbf{x},\boldsymbol{\theta}^{(0)})=\sum_{k=1}^{K}P(D=d|\mathbf{x},\mathbf{w}_k^{(0)})P(k|\mathbf{x},\mathbf{a}^{(0)}) \tag{7.19}$$

367

其中，$\boldsymbol{\theta}^{(0)}$ 是产生模型的参数向量，代表 $\mathbf{a}^{(0)}$ 和 $\{\mathbf{w}_k^{(0)}\}_{k=1}^{K}$ 的结合。在 $\mathbf{a}^{(0)}$ 和 $\mathbf{w}_k^{(0)}$ 中的上标 0 是用来区分产生模型的参数和下面要讨论的混合专家模型的参数的。

混合专家模型

考虑如图 7-8 所示的网络设置，称为混合专家(mixture of experts, ME)模型[4]。特别地，它由 K 个叫专家网络或是简称专家的监督模块组成，并且有一个叫门网(gating network)的整合单元，在专家网络中充当协调者的角色。假定不同的专家根据前面所讲的概率产生模型在输入空间不同的区域上工作得最好，这就需要门网协调。

将回归问题假定为是标量的，每一个专家网络包含一个线性滤波器。图 7-9 构成专家 k 的单个神经元的信号流图。因此，专家 k 产生的输出是输入向量 $\mathbf{x}$ 和该神经元突触权值向量 $\mathbf{w}_k$ 的内积，表示为

$$y_k=\mathbf{w}_k^T\mathbf{x}, \qquad k=1,2,\cdots,K \tag{7.20}$$

门网由单层的 K 个神经元组成，每个神经元被指派给一个特定的专家。图 7-10a 是门网的结构图，图 7-10b 是在该网络中神经元 k 的信号流图。和专家不一样，门网的神经元是非线性的，它们的激活函数由

$$g_k=\frac{\exp(u_k)}{\sum_{j=1}^{K}\exp(u_j)}, \qquad k=1,2,\cdots,K \tag{7.21}$$

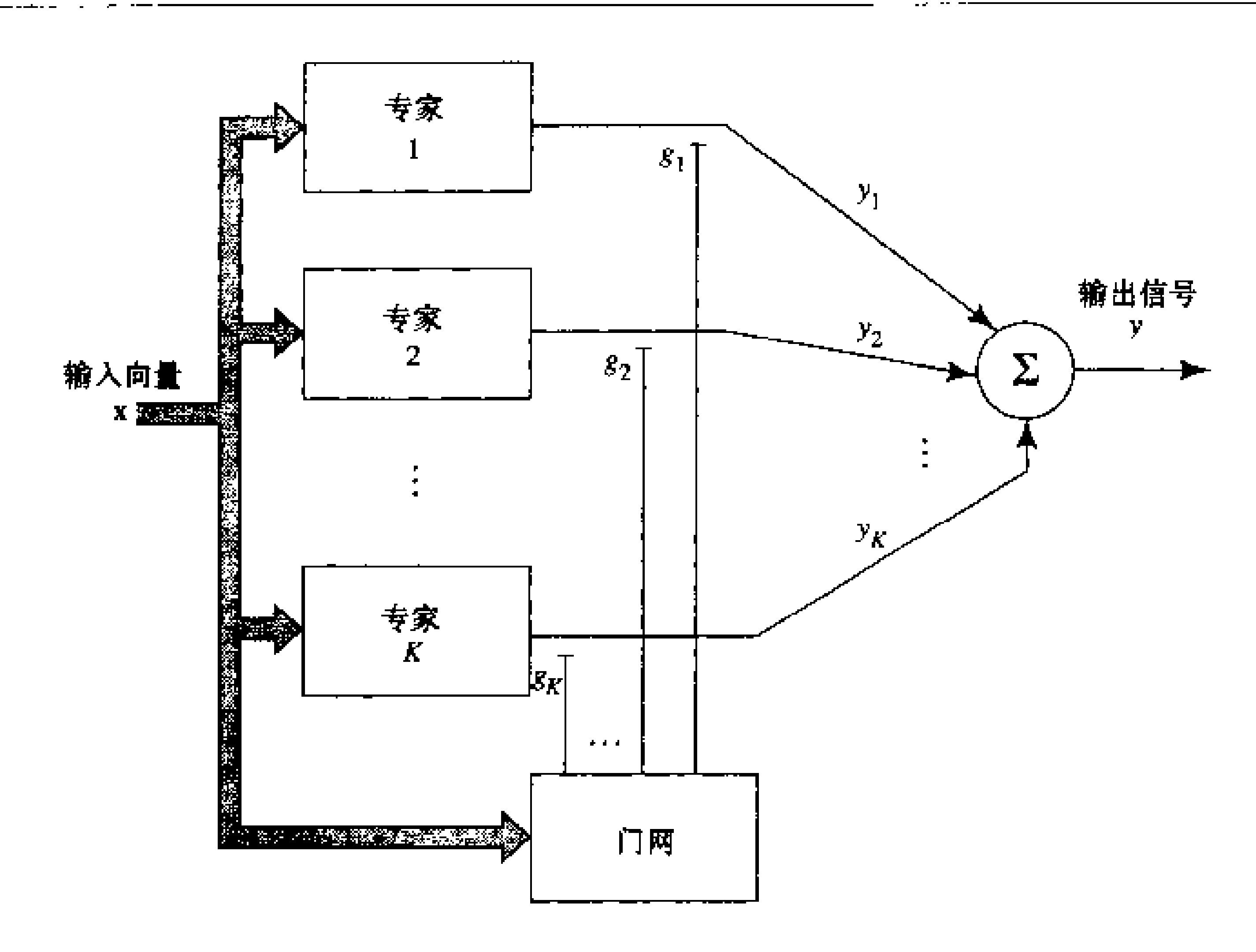
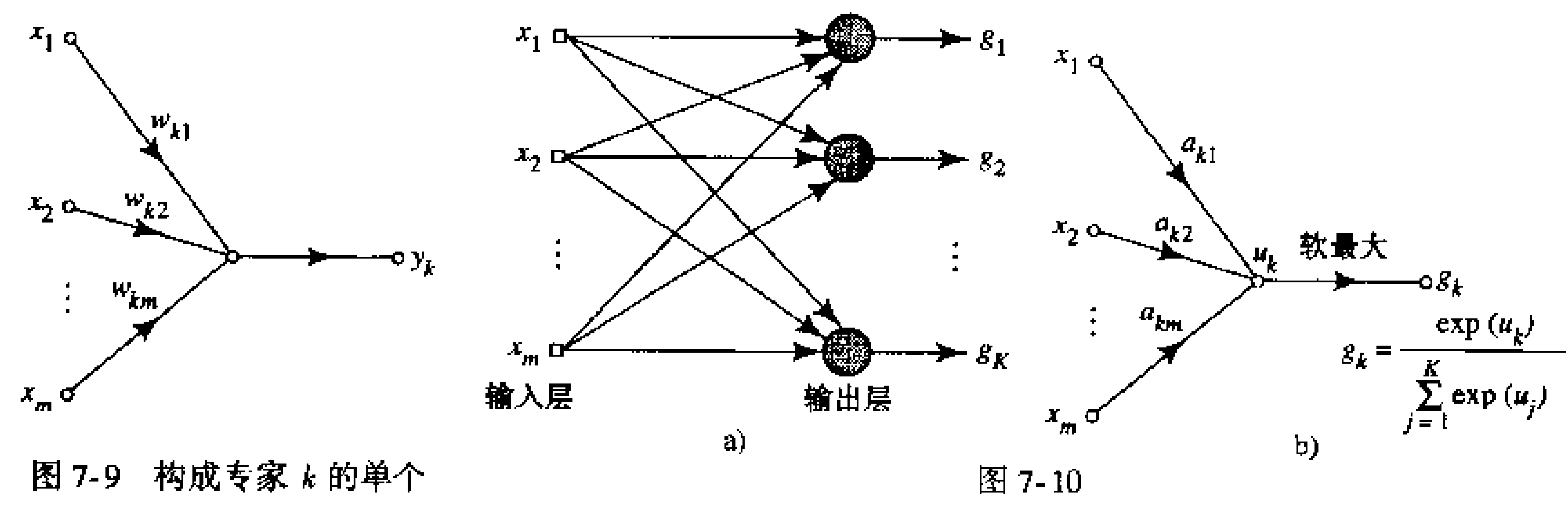


图 7-8 ME 模型的框图；门网协调专家的标量输出



368 图 7-9 构成专家 k 的单个线性神经元的信号流图

图 7-10 a)用于门网的单层软最大神经元 b)软最大神经元信号流图

定义，其中 u_k 是输入向量 $\mathbf{x}$ 和突触权值向量 $\mathbf{a}_k$ 的内积，即

$$u_k = \mathbf{a}_k^T\mathbf{x}, \qquad k = 1,2,\cdots,K \tag{7.22}$$

式(7.21)归一化的指数变换可以看作 logistic 函数的多输入推广。它保持了输入值的级次，且是一个选取最大值的"胜者全得"运算的可微分推广。由于这个原因，(7.21)的激活函数称为软最大(softmax)(Bridle,1990a)。注意由于 u_k 对输入 $\mathbf{x}$ 的线性依赖使得门网的输出是 $\mathbf{x}$ 的一个非线性函数。

对于门网作用的概率解释，我们可以认为它是一个分类器，将输入向量 $\mathbf{x}$ 映射到多项概率(multinomial probability)，以便不同的专家将能够匹配期望的响应(Jordan and Jacobs,1995)。369 最重要的是，将"软最大"用作门网的激活函数能确保这些概率满足以下要求：

$$0 \leq g_k \leq 1 \qquad 对于所有的\ k \tag{7.23}$$

和

$$\sum_{k=1}^{K} g_k = 1 \tag{7.24}$$

令 y_k 代表输入向量为 $\mathbf{x}$ 时第 k 个专家的输出。这个 ME 模型的整体输出是

$$y = \sum_{k=1}^{K} g_k y_k \tag{7.25}$$

其中，正像前面指出的那样，g_k 是 $\mathbf{x}$ 的一个非线性函数。当选定了概率产生模型的规则 k，单个输出 y_k 可以看作随机变量 D 的条件均值，表示为

$$E[D \mid \mathbf{x}, k] = y_k = \mathbf{w}_k^T \mathbf{x}, \qquad k = 1,2,\cdots,K \tag{7.26}$$

用 μ_k 表示 D 的条件均值，可以写成

$$\mu_k = y_k, \qquad k = 1,2,\cdots,K \tag{7.27}$$

D 的方差同误差 ϵ_k 的方差一样。因此根据式(7.18)，可以写出

$$\mathrm{var}[D \mid \mathbf{x}, k] = 1, \qquad k = 1,2,\cdots,K \tag{7.28}$$

当给定输入向量 $\mathbf{x}$ 和选取概率产生模型的第 k 个规则(即专家 k)后，D 的概率密度函数可以描述为

$$f_D(d \mid \mathbf{x}, k, \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(d - y_k)^2\right), \qquad k = 1,2,\cdots,K \tag{7.29}$$

其中 $\boldsymbol{\theta}$ 表示门网的参数和 ME 模型中那些专家的参数的参数向量。给定 $\mathbf{x}$，D 的概率密度函数是概率密度函数 $\{f_D(d \mid \mathbf{x}, k, \boldsymbol{\theta})\}_{k=1}^{K}$ 的混合，它的混合参数由门网决定的多项概率给出。因此可以写成

$$f_D(d \mid \mathbf{x}, \boldsymbol{\theta}) = \sum_{k=1}^{K} g_k f_D(d \mid \mathbf{x}, k, \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{K} g_k \exp\left(-\frac{1}{2}(d - y_k)^2\right) \tag{7.30}$$

370

式(7.30)的概率分布称为联想 Gauss 混合模型(associative Gaussian mixture model)，其非联想的对应物是传统 Gauss 混合模型(Titterington et al., 1985; McLachlan and Basford, 1988)，这在第 5 章简要描述。一个联想模型区别于非联想模型的不同之处在于其条件均值 μ_k 和混合参数 g_k 是非固定的；相反，它们都是输入向量 $\mathbf{x}$ 的函数。式(7.30)的联想 Gauss 混合模型可以被看作传统 Gauss 模型的推广。

图 7-8 所示 ME 模型假定通过训练得到恰当调整，则其重要方面是：

1. 给定 $\mathbf{x}$ 和概率产生模型的规则 k 成立，第 k 个专家的输出 y_k 提供代表期望响应 D 的随机变量的条件均值的一个估计。

2. 门网的输出 g_k 定义在单独从 $\mathbf{x}$ 获得知识的基础上专家 k 的输出匹配值 $D = d$ 的多项概率。

给定训练样本 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ 和以式(7.30)的概率分布工作，问题就是要以最优的方式学习条件均值 $\mu_k = y_k$ 和混合参数 g_k，$k = 1,2,\cdots,K$，使得 $f_D(d \mid \mathbf{x}, \boldsymbol{\theta})$ 提供负责产生训练数据的环境的固有概率密度函数的良好估计。

例 7.1　回归曲面　考虑一个包含两个专家和一个由 g_1 和 g_2 表示两个输出的门网的 ME 模型。输出 g_1 定义为(参看式(7.21))

$$g_1 = \frac{\exp(u_1)}{\exp(u_1) + \exp(u_2)} = \frac{1}{1 + \exp(-(u_1 - u_2))} \tag{7.31}$$

令 $\mathbf{a}_1$ 和 $\mathbf{a}_2$ 代表门网的两个权值向量。我们可以写成

$$u_k = \mathbf{x}^T \mathbf{a}_k, \qquad k = 1,2$$

从而重新将等式(7.31)写成

371

$$g_1 = \frac{1}{1 + \exp(-\mathbf{x}^T(\mathbf{a}_1 - \mathbf{a}_2))} \tag{7.32}$$

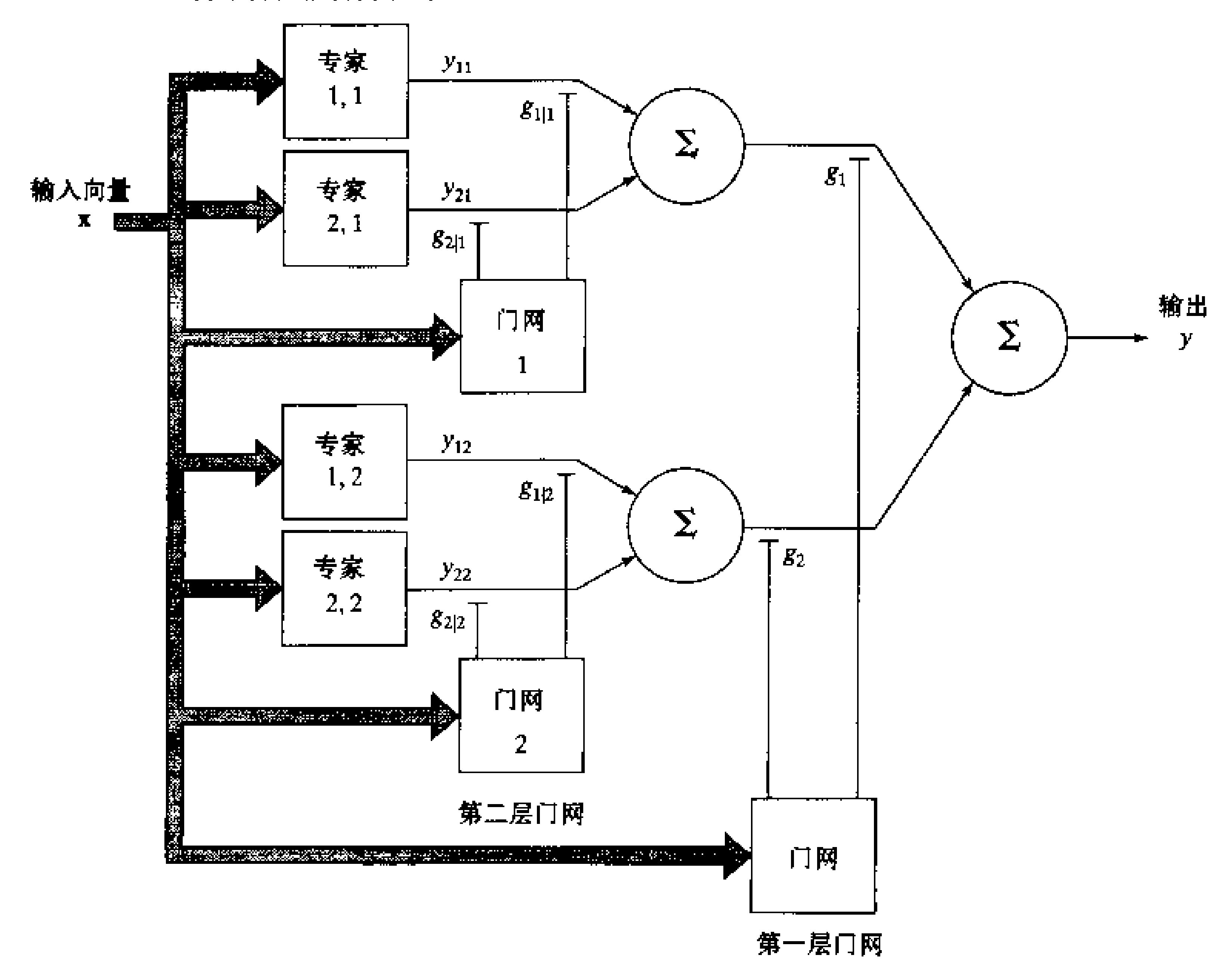
门网的另外一个输出 g_2 是

$$g_2 = 1 - g_1 = \frac{1}{1 + \exp(-\mathbf{x}^T(\mathbf{a}_2 - \mathbf{a}_1))}$$

因此，g_1，g_2 都是 logistic 函数的形式，但有一点差别。g_1 的方向由差向量($\mathbf{a}_1 - \mathbf{a}_2$)的方向决定，而 g_2 的方向由差向量($\mathbf{a}_2 - \mathbf{a}_1$)的方向决定，刚好和门 g_1 的方向相反。沿着由 $\mathbf{a}_1 = \mathbf{a}_2$ 定义的脊线，我们可以得到 $g_1 = g_2 = 1/2$，这两个专家对该 ME 模型的输出贡献是相同的。远离脊线，则这两个专家中的一个或者另外一个充当支配角色。

7.7 分层混合专家模型

如图 7-8 所示的 ME 模型的工作是通过将输入空间分解成不同的子空间，由一个门网负责分散信息(从训练数据中收集)给不同的专家。如图 7-11 所示的*分层混合专家*(HME)模型是 ME 模型的自然扩展。这个图例是由四个专家组成的一个 HME 模型。HME 模型的体系结构是一棵树，门网在树的非终端节点，而专家在树的叶子部分。HME 模型和 ME 模型的不同之处在于其输入空间被分成一个*嵌套*的子空间集，在多个以*分层*方式调整的门网控制下信息在专家之间被整合或者重新分配。



372

图 7-11 两个层次的层次混合专家(HME)示意图

如图 7-11 所示的 HME 模型有*两层层次或两层门网*。继续以同样方式运用分而治之的原则，我们可以构造任意多层层次的 HME 模型。注意根据图 7-11 所描述的约定，门网层的编号从树的输出节点开始。

图 7-11 所示的 HME 模型的构成可以从两方面观察(Jordan,1994)：

1. *HME 模型是分而治之策略的产物*。如果我们相信将输入空间分成区域是一个好策略，那么再将区域分成子区域是一个同样好的策略。我们可以递归地继续采用这种方式，直到达到这样一个阶段，逼近曲面的复杂性是对训练数据"局部"复杂性良好拟合。因此 HME 模型至少应有 ME 模型一样的性能，而且经常要比它好。这是基于这样原因：一个 HME 模型中较高层的门网有效地整合信息，并且把它重新分配给该门网控制下的特定子树的专家。因此，在所讨论的子树中每一个参数和在该子树中的其他参数一起分享强度，因而有助于提高 HME 模型的整体性能。

2. *HME 模型是一个软决策树*。根据这种观点，混合专家只不过是单层的决策树，有时也称为*决策树桩*(decision stump)。从一个更一般的背景来说，HME 模型可视为决策树的概率框架，具有称为决策树树根的 HME 模型的输出节点。标准决策树的方法是构造一棵树，该树在输入空间的不同域上导出一个硬(即是或否)决策。这和 HME 模型上的软决策形成对照。因此，基于下面的两个原因 HME 模型会胜过标准决策树：

- 一个硬决策不可避免的丢失信息，但一个软决策树尽力地保存信息。例如一个软二分决策传送距决策边界(即其决策是 0.5 的点)的距离信息，而一个硬决策做不到这一点。因此我们可以说不像标准的决策树，HME 模型符合*信息保持规则*(information preservation rule)。这个经验规则表明一个输入信号的信息内容应该以计算有效的方式保存直到系统作好进行最后决策或者参数估计的准备。
- 标准决策树受到*贪婪*(greediness)问题的损害。一旦从这样的树中作出一个决策，那么在这以后这个决策被冻结，永久不会改变。HME 模型减轻了贪婪问题，因为通过这棵树所作的决策是不断变化的。不像标准决策树，在 HME 模型中不良决策可能沿着这棵树得到恢复。

第二种观点，即在考虑 HME 模型时一个软决策树是首选的方法。当将 HME 模型看作决策树的概率基础时，对任何给定的数据集它允许我们计算似然函数，并且对决定输入空间不同区域之间分割的参数求最大似然估计。因而在我们已知的标准决策树的基础上，可以得到一个实际的模型选择问题的解决方案，这在下一节进行讨论。 373

7.8 使用标准决策树的模型选择

和每一种其他的神经网络一样，对于参数估计问题的一个满意解，关键在于对所解决的问题选择合适的模型。在 HME 模式的情形，模型选择包括树中的决策节点的数目和组织。这种特殊的模型选择问题的一个确实可行的解决方案是在训练集上运行标准决策树算法，然后采用获得的树作为决定 HME 模型的参数的学习算法的初始化步骤(Jordan,1994)。

HME 模型和标准决策树有很清晰的相似性，比如 Breiman et al.(1984)提出的*分类和回归树*(classification and regression tree,CART)。图 7-12 表示一个 CART 的例子，其中输入数据的空间$\mathscr{X}$被一系列的二值分割剖分成终端节点。比较图 7-11 和图 7-12，我们会发现 CART 和 HME 之间的下述相似点：

- 在 CART 的中间(即非终端)节点中选择分割的规则所起的作用，和 HME 模型中门网的作用相似。
- CART 中的终端节点所起的作用，和 HME 模型中专家网络作用相似。

从对感兴趣的分类或回归问题的 CART 开始，我们利用 CART 的离散性，在可选择树中提供一种有效的搜索。通过应用这样选择的一棵树作为参数估计学习算法的初始化步骤，我们利用 HME 模型的连续概率基础产生期望响应的一个改进的"软"估计。

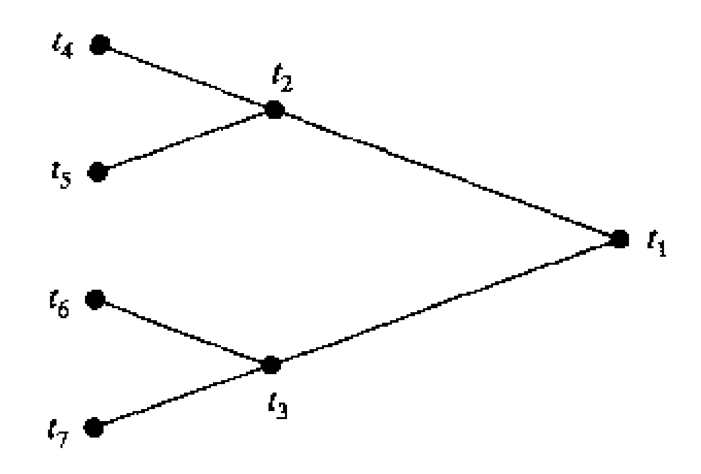


图 7-12 二叉决策树，描述如下：节点 t_2 和节点 t_3 为节点 t_1 的后代；节点 t_4 和节点 t_5 为节点 t_2 的后代；节点 t_6 和节点 t_7 为节点 t_3 的后代

CART 算法

根据我们刚才所讲的，可以得到一个 CART 算法的简明描述。该描述在回归的背景下给出。以训练数据 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ 开始，我们可以利用 CART 通过以下的方式去建造一个最小平方回归的二叉树 T(Breiman et al.,1984)：

1. 分割的选择。设一个节点 t 代表当前树 T 的一个子集。让 $\bar{d}(t)$ 代表所有落入 t 的 $(\mathbf{x}_i, d_i)$ 的 d_i 平均，即

$$\bar{d}(t) = \frac{1}{N(t)} \sum_{\mathbf{x}_i \in t} d_i \tag{7.33}$$

其中，$N(t)$ 是 t 中所有实例的数目，对所有 $\mathbf{x}_i \in t$ 的 d_i 求和。定义

374

$$\mathscr{E}(t) = \frac{1}{N} \sum_{\mathbf{x}_i \in t} (d_i - \bar{d}(t))^2 \tag{7.34}$$

和

$$\mathscr{E}(T) = \sum_{t \in T} \mathscr{E}(t) \tag{7.35}$$

对于节点 t，总和 $\sum_{\mathbf{x}_i \in t} (d_i - \bar{d}(t))^2$ 代表"节点内的平方和"，即它是所有的在 t 中的 d_i 和均值 $\bar{d}(t)$ 的偏差平方总和。将这些 $t \in T$ 的偏差加起来得到所有节点的偏差的平方之和，被 N 除后得到均值。

给定 T 中当前节点 t 的一个分割集 S，最好的分割 s^* 是 S 中使 $\mathscr{E}(T)$ 减少最快的分割。更精确的说，假定对于节点 t 的任何分割 s，它将节点 t 分成 t_L(t 左边的新节点)和 t_R(t 右边的新节点)，我们令

$$\Delta \mathscr{E}(s,t) = \mathscr{E}(T) - \mathscr{E}(t_L) - \mathscr{E}(t_R) \tag{7.36}$$

那么要采取的最好分割 s^* 是一个如下的特殊分割

$$\Delta \mathscr{E}(s^*,t) = \max_{s \in S} \Delta \mathscr{E}(t,s) \tag{7.37}$$

建立一棵回归树以使 $\mathscr{E}(T)$ 的减少最大化。

2. 终端节点的确定。假如下面的条件满足，一个 t 节点被声明为终端节点：

$$\max_{s \in S} \Delta \mathscr{E}(s,t) < \beta \tag{7.38}$$

其中 β 为预先给出的阈值。

3. 终端节点参数的最小平方估计。令 t 代表最后的二叉树 T 的终端节点，令 $\mathbf{X}(t)$代表由 $\mathbf{x}_i \in t$ 组成的矩阵。令 $\mathbf{d}(t)$代表 t 中所有 d_i 组成的对应向量，定义

$$\mathbf{w}(t) = \mathbf{X}^{+}(d)\mathbf{d}(t) \tag{7.39}$$

其中，$\mathbf{X}^{+}(t)$是矩阵 $\mathbf{X}(t)$的伪逆。应用 $\mathbf{w}(t)$将在终端节点 t 输出中产生一个 $d(t)$的最小平方估计。使用式(7.39)计算产生的权值，通过寻找关于回归曲面残差(误差)平方的最小和而不是均值，分割选择问题得以求解。

使用 CART 初始化 HME 模型

假定 CART 的方法已经应用到一个训练集上，由此产生这个问题的一个二叉决策树。我们可以把由 CART 产生的分割作为一个多维曲面，定义为 [375]

$$\mathbf{a}^T\mathbf{x} + b = 0$$

其中，$\mathbf{x}$ 是输入向量，$\mathbf{a}$ 代表一个参数向量，b 代表一个偏置。

接着考虑一个 HME 模型中对应的情况，从例 7.1 我们注意在一个二叉树中由一个门网产生的回归曲面可以写成

$$g = \frac{1}{1 + \exp(-(\mathbf{a}^T\mathbf{x} + b))} \tag{7.40}$$

它定义一个分割，特别是 $g = 1/2$ 的时候。令这个特殊的门网的权值向量(差)$\mathbf{a}$ 被写为

$$\mathbf{a} = \|\mathbf{a}\| \cdot \frac{\mathbf{a}}{\|\mathbf{a}\|} \tag{7.41}$$

其中 $\|\mathbf{a}\|$ 代表 $\mathbf{a}$ 的长度(即欧几里德范数)，$\mathbf{a}/\|\mathbf{a}\|$ 是一个归一化的单位长度向量，将式(7.41)应用到式(7.40)中去，我们可以重写门网的一个参数化分割如下：

$$g = \frac{1}{1 + \exp\left(-\|\mathbf{a}\|\left(\left(\frac{\mathbf{a}}{\|\mathbf{a}\|}\right)^T\mathbf{x} + \frac{b}{\|\mathbf{a}\|}\right)\right)} \tag{7.42}$$

其中可以看出 $\mathbf{a}/\|\mathbf{a}\|$ 决定分割的方向，$\|\mathbf{a}\|$ 决定分割的锐度(sharpness)。通过第 2 章的讨论，我们观察到向量 $\mathbf{a}$ 的长度实际上充当温度的倒数。从式(7.42)中注意的重点是由线性过滤器后跟一个非线性的“软最大”形式组成的门网能够模仿一个 CART 类型的分割。此外，我们有另外的自由度，即向量 $\mathbf{a}$ 的长度。在一个标准决策树中，这个附加的参数是不相干的，因为用一个阈值(硬决策)来产生一个分割。相反，$\mathbf{a}$ 的长度对由 HME 模型中的门网产生的分割锐度有极深的影响。特别地，对于一个固定方向的突触权值向量 $\mathbf{a}$，我们可以陈述如下：

- 当 $\mathbf{a}$ 长(即温度低)的时候，分割是尖锐的；
- 当 $\mathbf{a}$ 短(即温度高)的时候，分割是柔和的。

假如在极限情况，我们有 $\|\mathbf{a}\| = 0$，分割消失并且在消失的(虚构的)分割两边 $g = 1/2$。因为被考虑的门网不再分割，所以设置 $\|\mathbf{a}\| = 0$ 的作用等同于从树中剪除非终端节点。在一个极端的例子中，当 $\|\mathbf{a}\|$ 在每一个非终端节点上很小(即温度高)，那么整个 HME 模型会像单个的节点；也就是说，HME 模型退化成一个线性回归模型(假设线性专家)。随着阈值的突触权值向量在长度上开始增加，HME 开始产生(软的)分割，因而增加模型可利用的自由度的数目。

我们可以通过如下的步骤初始化 HME：

1. 使用CART训练数据。

2. 设置HME模型中专家的突触权值向量，使其和在二叉树相应终端节点上通过应用CART得到的参数向量最小平方估计相等。

376

3. 对于门网：

(a)设置突触权值向量，使其指向二叉树中与通过CART得到的相应分割正交的方向。

(b)设置突触权值向量的长度(即欧几里德范数)等于很小的随机向量。

7.9 先验和后验概率

多项式概率 g_k 和 $g_{j|k}$ 分别属于第一层和第二层的门网，从它们的值仅依赖于输入向量(刺激)**x**这个意义上来说，可视为先验概率。用同样的方法，可以定义后验概率 $h_{j|k}$ 和 h_k，它们的值既依赖于输入向量**x**，又依赖于专家对**x**的响应。后面的这组概率对HME模型的学习方法的发展有用。

参考图7-11的HME模型，可以定义树中非终端节点的后验概率为(Jordan and Jacobs, 1994)：

$$h_k = \frac{g_k \sum_{j=1}^{2} g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)}{\sum_{k=1}^{2} g_k \sum_{j=1}^{2} g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)} \tag{7.43}$$

和

$$h_{j|k} = \frac{g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)}{\sum_{j=1}^{2} g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)} \tag{7.44}$$

h_k 和 $h_{j|k}$ 的乘积定义专家(j,k)所产生输出 y_{jk} 匹配期望响应 d 的联合后验概率，由

$$h_{jk} = h_k h_{j|k} = \frac{g_k g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)}{\sum_{k=1}^{2} g_k \sum_{j=1}^{2} g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right)} \tag{7.45}$$

给出。概率 h_{jk} 满足下面的两个条件：

$$0 \leqslant h_{jk} \leqslant 1 \qquad \text{对于所有的}(j,k) \tag{7.46}$$

377

$$\sum_{j=1}^{2} \sum_{k=1}^{2} h_{jk} = 1 \tag{7.47}$$

式(7.47)的含义为信任是在竞争的基础上在专家之间分配。此外，从式(7.45)注意到，y_{jk} 与 d 越接近，给予专家(j,k)的输出匹配 d 的信任就越多，这是直观上满足的。

HME模型的一个特别值得一提的重要特征是计算后验概率涉及的计算递归性。检查式(7.42)和(7.43)，发现式(7.44)中 $h_{j|k}$ 的分母看起来是式(7.43)中 h_k 的分子。在一个HME模型中，我们想计算树中所有非终端节点的后验概率。这正是递归性特别有价值之处。特别地，计算树中的所有非终端节点的后验概率可以通过如下描述的一遍过程得到：

- 从这棵树一层一层地移动到根节点，树的所有非终端节点可以通过简单地将它的"孩子们"的后验概率进行整合而得到。

7.10 最大似然估计

下面转向HME模型的参数估计问题，我们首先注意它的概率的解释和ME模型有某些不同。因为HME模型以二叉树的形式组织起来，所以假定负责产生数据的环境包括一个嵌套序列的软(二叉)决策，在输入向量**x**到输出d的回归中结束。特别地，我们假定在HME的概率产生模型中，决策模拟为多项式随机变量(Jordan and Jacobs, 1994)。即对于每一个输入**x**，我们将$g_i(\mathbf{x},\boldsymbol{\theta}_i^0)$解释为和第一个决策有关的多项式概率，将$g_{j|i}(\mathbf{x},\boldsymbol{\theta}_{ji}^0)$解释为和第二个决策有关的条件多项式分布。和前面的一样，上标0表示产生模型参数的真实值。这个决策形成一个决策树。和ME模型一样，"软最大"被用作整个HME模型的门网的激活函数。特别地，顶层门网的第k个输出神经元的激活g_k如下定义：

$$g_k = \frac{\exp(u_k)}{\exp(u_1) + \exp(u_2)}, \qquad k = 1,2 \tag{7.48}$$

其中u_k是应用到那个神经元的输入加权和。类似地，第二层第k个门网的第j个输出神经元的激活定义为

$$g_{j|k} = \frac{\exp(u_{jk})}{\exp(u_{1k}) + \exp(u_{2k})}, \qquad (j,k) = 1,2 \tag{7.49}$$

[378]

其中u_{jk}是应用到这个特定神经元的输入加权和。

由于表示的原因，我们将要讲到的HME模型仅仅只有两层层次(即两层门网)，如图7-11所示。和ME模型一样，HME模型的每一个专家被假定为由一个单层的线性神经元组成。令y_{jk}代表专家(j,k)的输出，可以把HME模型的整体输出表示为

$$y = \sum_{k=1}^{2} g_k \sum_{j=1}^{2} g_{j|k} y_{jk} \tag{7.50}$$

遵循类似于7.6节描述用于ME模型的过程，给定输入**x**，我们可以对图7-11的HME模型的期望响应的随机变量D的概率密度函数表示如下：

$$f_D(d \mid \mathbf{x},\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{2} g_k \sum_{j=1}^{2} g_{j|k} \exp\left(-\frac{1}{2}(d - y_{jk})^2\right) \tag{7.51}$$

因而，对于一个给定的训练数据集，式(7.51)定义一个数据的固有分布的模型。向量$\boldsymbol{\theta}$包括HME模型中表征门网和专家网络涉及的所有突触权值。

似然函数$l(\boldsymbol{\theta})$的设计由概率函数$f_D(d|\mathbf{x},\boldsymbol{\theta})$给出，可看作一个参数向量$\boldsymbol{\theta}$的函数。因此我们可以写成

$$l(\boldsymbol{\theta}) = f_D(d \mid \mathbf{x},\boldsymbol{\theta}) \tag{7.52}$$

虽然条件联合概率密度函数和似然函数是同样的公式，但我们必须理解它们的不同之处。在$f_D(d|\mathbf{x},\boldsymbol{\theta})$中，输入向量**x**和参数向量$\boldsymbol{\theta}$是固定的，而期望响应$d$是变量。但是，在似然函数$l(\boldsymbol{\theta})$中，**x**和$d$都是固定的，而$\boldsymbol{\theta}$是变量。

实际上，我们发现似然函数的自然对数使用起来比似然函数本身方便得多。用$L(\boldsymbol{\theta})$表示对数似然函数，写成

$$L(\boldsymbol{\theta}) = \log[l(\boldsymbol{\theta})] = \log[f_D(d \mid \mathbf{x},\boldsymbol{\theta})] \tag{7.53}$$

$l(\boldsymbol{\theta})$的自然对数为$l(\boldsymbol{\theta})$的单调变换。这意味着$l(\boldsymbol{\theta})$只要增加，其自然对数$L(\boldsymbol{\theta})$也增加。因为$l(\boldsymbol{\theta})$是一个条件概率密度函数的公式，它永远不可能为负。那就意味着求$L(\boldsymbol{\theta})$的计算无

任何问题。因此参数向量 $\boldsymbol{\theta}$ 的一个估计值 $\hat{\boldsymbol{\theta}}$ 能通过似然方程

$$\frac{\partial}{\partial \boldsymbol{\theta}} l(\boldsymbol{\theta}) = 0$$

得到，或者等价地从对数似然方程

379

$$\frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta}) = 0 \tag{7.54}$$

得到。具有所期望的渐进性质[5]的“最大似然估计”的术语通常是指能使似然函数 $l(\boldsymbol{\theta})$ 达到全局最大化的似然函数方程的根。但是，实际使用的估计值 $\hat{\boldsymbol{\theta}}$，事实上可能是局部最大而不是全局最大。无论如何，归功于 Fisher(1925)的最大似然估计，基于一个相对简单的思想：

不同的总体产生不同的数据样本，并且任何一个给定的数据样本更有可能从某个总体而不是从其他的总体产生。

更确切地说，给定输入向量 $\mathbf{x}$，未知参数向量 $\boldsymbol{\theta}$ 是通过它的最可能值估计的。换句话说，最大似然估计 $\hat{\boldsymbol{\theta}}$ 是使得其条件概率函数 $f_D(d|\mathbf{x},\boldsymbol{\theta})$ 最大的参数向量 $\boldsymbol{\theta}$ 的值。

7.11 HME 模型的学习策略

7.10 节中 HME 模型的概率描述引导我们将对数似然函数 $L(\boldsymbol{\theta})$ 作为最大化的目标函数。此时关键问题是如何实现最大化。和其他最优化问题一样，并不是只有独一无二的最大化 $L(\boldsymbol{\theta})$ 的方法。相反，我们有好几个达到我们目的的方法，在这里概述其中的两个(Jacobs and Jordan，1991；Jordan and Jacobs，1994)：

1. 随机梯度方法。这个方法产生 $L(\boldsymbol{\theta})$ 的最大化的在线算法。对于如图 7-11 描述的两层 HME 模型依赖于下面组成的公式：

- 专家(j,k)中突触权值向量的梯度向量$\partial L/\partial \mathbf{w}_{jk}$
- 顶层门网中输出神经元 k 的突触权值向量的梯度向量$\partial L/\partial \mathbf{a}_k$
- 和专家(j,k)相连的第二层门网中输出神经元的突触权值向量的梯度向量$\partial L/\partial \mathbf{a}_{jk}$

下面的公式可直接证明：

$$\frac{\partial L}{\partial \mathbf{w}_{jk}} = h_{j|k}(n)h_k(n)(d(n) - y_{jk}(n))\mathbf{x}(n) \tag{7.55}$$

$$\frac{\partial L}{\partial \mathbf{a}_k} = (h_k(n) - g_k(n))\mathbf{x}(n) \tag{7.56}$$

380

$$\frac{\partial L}{\partial \mathbf{a}_{jk}} = h_k(n)(h_{j|k}(n) - g_{j|k}(n))\mathbf{x}(n) \tag{7.57}$$

式(7.55)表明，在训练的过程中，对专家(j,k)的突触权值的调整，是与联合后验概率 h_{jk}成比例地修正输出y_{jk}和期望响应 d 之间的误差。式(7.56)表明，对顶层门网的输出神经元 k 的突触权值的调整，是使得后验概率 $g_k(n)$和相应的后验概率 $h_k(n)$逐渐靠近。式(7.57)表明，对与专家(j,k)相联系的第二层门网输出神经元的突触的调整，是与后验概率 $h_k(n)$成比例地修正先验概率 $g_{j|k}$和后验概率 $h_{j|k}$之间的误差。

根据式(7.55)至式(7.57)，当每一个模式(刺激)被出现后，HME 模型的突触权值要相应地更新。通过将梯度向量对 n 求和，可以得到使对数似然函数 $L(\boldsymbol{\theta})$最大化的集中式的梯度

上升算法。

2. *期望最大化方法*。*期望最大化*(expectation-maximization, EM)算法归功于 Dempster et al. (1977)，提供一个在有缺失数据情况下计算最大似然估计值的迭代方法，在此情况下如果没有数据缺失，则最大似然估计将是一件简单的事情。EM 算法的名字是根据在该算法的每次迭代中都有两步这个事实而得来的：

- *期望步*或者是 E 步，它使用一个*非完整数据*(*incomplete data*)问题的观察数据集和参数向量的当前值，产生一个假定的扩大的或者称为完整的数据集。
- *最大化步*或者 M 步，它通过使 E 步产生的完整数据的对数似然函数最大化导出参数向量的一个新的估计值。

因此，参数向量从一个合适的值开始，E 步和 M 步交替进行直到收敛。

EM 算法适用的情况不仅仅包括那些本来就非完整的数据，还包括其他各种不同情况，这些情况下数据非完整对讨论的问题而言一点也不明显或者说不自然。实际上，最大似然估计的计算通过人工地使它成为不完整数据问题经常极其容易。之所以这样是因为 EM 算法在给定完整数据的情况下能有效利用减低后的最大似然估计的复杂性(McLachlan and Krishnan, 1997)。HME 模型是这样的应用例子之一。在这种情况下，缺失数据以某种指示器变量的形式人工地引入到 HME 模型中，以方便估计未知参数向量的最大似然值，正如在 7.12 节讨论过的一样。

不管是通过随机梯度方法还是应用 ME 算法进行设计，HME 模型的重要特征是双重的：

- 模型中的每一个门网不断地计算训练集的每个数据点的后验概率。
- 应用于模型中专家和门网的突触权值的调整量，从一次迭代到下一次，是一个所计算的后验概率和相应的先验概率的函数。

相应的，假如树底部的专家网络不能很好地拟合其局部邻域的训练数据，那么树中高层的门网的回归(判别)曲面将被移向周围。这种移动反过来能帮助专家网络在下一次学习算法的迭代中通过平移它们进行数据拟合的子空间而更好地拟合数据。HME 模型就是通过这种过程来改良与像 CART 这样的标准决策树有关的贪婪问题。 381

7.12 EM 算法

EM 算法之所以值得注意，部分是由于固有理论的简单性和通用性，部分由于其广泛的运用[6]。在这一节我们将在一般意义下对 EM 算法做一个简单的描述。在下一节我们继续考虑它在 HME 模型的参数估计问题中的应用。

让向量 $\mathbf{z}$ 代表缺失的或者未观察到的数据。让 $\mathbf{r}$ 代表完整的数据向量，它由一些可观察的数据 $\mathbf{d}$ 和缺失的数据向量 $\mathbf{z}$ 组成。因而考虑两个数据空间$\mathcal{R}$和$\mathcal{D}$，它们具有从$\mathcal{R}$到$\mathcal{D}$的多对一的映射。我们不能观察到完整的数据向量 $\mathbf{r}$，相反实际仅能观察到$\mathcal{D}$中非完整的数据 $d = d(\mathbf{r})$。

令 $f_c(\mathbf{r}|\boldsymbol{\theta})$代表在给定参数向量 $\boldsymbol{\theta}$ 的情况下 $\mathbf{r}$ 的条件概率密度函数。那么随机变量 D 在给定 $\boldsymbol{\theta}$ 的情况下的条件概率密度函数可以定义为

$$f_D(d \mid \boldsymbol{\theta}) = \int_{\mathcal{R}(d)} f_c(\mathbf{r} \mid \boldsymbol{\theta})\, d\mathbf{r} \tag{7.58}$$

其中$\mathcal{R}(d)$由 $d = d(\mathbf{r})$决定的$\mathcal{R}$的子空间。EM 算法的直接目的在于找到 $\boldsymbol{\theta}$ 的一个值使得非完

整数据的对数似然函数

$$L(\boldsymbol{\theta}) = \log f_D(d \mid \boldsymbol{\theta})$$

取得最大。但是，这个问题的解决是通过间接地运用完整数据的对数似然函数

$$L_c(\boldsymbol{\theta}) = \log f_c(\mathbf{r} \mid \boldsymbol{\theta}) \tag{7.59}$$

进行迭代来完成的，它是一个随机变量，因为缺失数据向量 $\mathbf{z}$ 是未知的。

更确切地说，让 $\hat{\boldsymbol{\theta}}(n)$ 代表 EM 算法在迭代 n 时参数向量 $\boldsymbol{\theta}$ 的值。在这次迭代的 E 步，我们计算期望

$$Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(n)) = E[L_c(\boldsymbol{\theta})] \tag{7.60}$$

其中期望是对 $\hat{\boldsymbol{\theta}}(n)$ 得到的。在同一的迭代的 M 步，在参数(权值)空间 $\mathscr{W}$ 中对 $\boldsymbol{\theta}$ 最大化 $Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(n))$，这样找到更新参数估计值 $\hat{\boldsymbol{\theta}}(n+1)$，表示为

$$\hat{\boldsymbol{\theta}}(n+1) = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(n)) \tag{7.61}$$

该算法开始时参数向量 $\boldsymbol{\theta}$ 的初始值为 $\hat{\boldsymbol{\theta}}(0)$，然后根据式(7.60)和(7.61)交替进行 E 步和 M 步，直到 $L(\hat{\boldsymbol{\theta}}(n+1))$ 和 $L(\hat{\boldsymbol{\theta}}(n))$ 之间的差下降至某一任意小值；此时，整个计算结束。

注意在 EM 算法的一次迭代后，非完整数据对数似然函数不是递减的，表示为(参看习题 7.10)

382

$$L(\hat{\boldsymbol{\theta}}(n+1) \geq L\hat{\boldsymbol{\theta}}(n)), \qquad n = 0,1,2,\cdots \tag{7.62}$$

等号成立意味着我们处于对数似然函数的稳定点[7]。

7.13 EM 算法在 HME 模型中的应用

在熟悉 EM 算法之后，我们准备应用 EM 算法解决 HME 模型的参数估计问题[8]。

考虑图 7-11 所示的 HME 模型，当它运行训练集的样本 i 时，令 $g_k^{(i)}$ 和 $g_{j|k}^{(i)}$ 分别代表第一层门网 k 和第二层门网 (j,k) 采取与决策有关的(条件)多项式概率。那么，我们很容易得到在给定样本 $\mathbf{x}_i$ 和参数向量 $\boldsymbol{\theta}$ 的情况下，随机变量 D 相应的条件概率密度函数的值如下：

$$f_D(d_i \mid \mathbf{x}_i, \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}} \sum_{k=1}^{2} g_k^{(i)} \sum_{j=1}^{2} g_{j|k}^{(i)} \exp\left(-\frac{1}{2}(d_i - y_{jk}^{(i)})^2\right) \tag{7.63}$$

其中，$y_{jk}^{(i)}$ 是为了响应训练集合的第 i 个样本由专家 (j,k) 产生的输出。假定包含在训练集内的所有 N 个样本彼此之间是统计独立的，对于非完整数据问题可以写出对数似然函数的公式

$$L(\boldsymbol{\theta}) = \log\left[\prod_{i=1}^{N} f_D(d_i \mid \mathbf{x}_i, \boldsymbol{\theta})\right] \tag{7.64}$$

利用式(7.63)代入式(7.64)且忽略常数 $-(1/2)\log(2\pi)$，可以得到

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log\left[\sum_{k=1}^{2} g_k^{(i)} \sum_{j=1}^{2} g_{j|k}^{(i)} \exp\left(-\frac{1}{2}(d_i - y_{jk}^{(i)})^2\right)\right] \tag{7.65}$$

为了计算 $\boldsymbol{\theta}$ 的最大似然估计值，我们不得不找一个 $L(\boldsymbol{\theta})$ 的稳定点(即局部或全局最大)。不幸的是，式(7.65)所示的最大似然函数 $L(\boldsymbol{\theta})$，并不能使我们很容易进行这种计算。

为了克服这种计算上的困难，根据 EM 算法我们通过加入一组相应的缺失数据人为地扩大可观察数据 $\{d_i\}_{i=1}^{N}$。为这一点引入属于 HME 结构概率模型的指示器变量如下：

- $z_k^{(i)}$ 和 $z_{j|k}^{(i)}$ 被解释为对训练集中第 i 个样本所做决策的相应标号。这些变量这样定义，

使得对于所有 i，只有一个 $z_k^{(i)}$ 等于 1，也只有一个 $z_{j|k}^{(i)}$ 等于 1。$z_k^{(i)}$ 和 $z_{j|k}^{(i)}$ 都是独立的离散随机变量，它们各自的期望定义为

$$E[z_k^{(i)}] = P[z_k^{(i)} = 1 \mid \mathbf{x}_i, d_i, \hat{\boldsymbol{\theta}}(n)] = h_k^{(i)} \tag{7.66}$$

$$E[z_{j|k}^{(i)}] = P[z_{j|k}^{(i)} = 1 \mid \mathbf{x}_i, d_i, \hat{\boldsymbol{\theta}}(n)] = h_{j|k}^{(i)} \tag{7.67}$$

[383]

其中，$\hat{\boldsymbol{\theta}}(n)$ 是参数向量 $\boldsymbol{\theta}$ 在 EM 算法迭代 n 次时的估计。

- $z_{jk}^{(i)} = z_k^{(i)} z_{j|k}^{(i)}$ 被解释成对训练集中第 i 个例子指定概率模型的专家 (j,k) 的标号，它也被看作一个离散的随机变量，其期望值定义为

$$E[z_{jk}^{(i)}] = E[z_{j|k}^{(i)} z_k^{(i)}] = E[z_{j|k}^{(i)}]E[z_k^{(i)}] = h_{j|k}^{(i)} h_k^{(i)} = h_{jk}^{(i)} \tag{7.68}$$

式(7.66)至(7.68)中的 $h_k^{(i)}$，$h_{j|k}^{(i)}$ 和 $h_{jk}^{(i)}$ 是 7.9 节引入的后验概率；对它们添加上标 i 表明当前考虑的样本。这三个等式的合理性参看习题 7.13。

通过将如此定义的缺失数据加入到可观察数据中，最大似然估计问题被大大地简化了。更确切地说，在给定了 $\mathbf{x}_i$ 和参数向量 $\boldsymbol{\theta}$ 的情况下，令 $f_c(d_i, z_{jk}^{(i)} \mid \mathbf{x}_i, \boldsymbol{\theta})$ 代表由 d_i 和 $z_{jk}^{(i)}$ 组成的完整数据的条件概率密度函数，我们可以写成

$$f_c(d_i, z_{jk}^{(i)} \mid \mathbf{x}_i, \boldsymbol{\theta}) = \prod_{j=1}^{2} \prod_{k=1}^{2} (g_k^{(i)} g_{j|k}^{(i)}(i) f_{jk}(d_i)) \tag{7.69}$$

其中 $f_{jk}(d_i)$ 是在给定选择 HME 模型专家 (j,k) 的情况下 d_i 的条件概率密度函数，$f_{jk}(i)$ 由 Gauss 分布

$$f_{jk}(d_i) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(d_i - y_{jk}^{(i)})^2\right) \tag{7.70}$$

给出。注意公式(7.69)对应于一个假想实验，它含有由 $z_{jk}^{(i)}$ 表示的实际不可观察的指示器变量。无论如何，完整数据问题的对数似然函数对应于整个训练集，由

$$\begin{aligned} L_c(\boldsymbol{\theta}) &= \log\left[\prod_{i=1}^{N} f_c(d_i, z_{jk}^{(i)} \mid \mathbf{x}_i, \boldsymbol{\theta})\right] = \log\left[\prod_{i=1}^{N} \prod_{j=1}^{2} \prod_{k=1}^{2} (g_k^{(i)} g_{j|k}^{(i)} f_{jk}(d_i))^{z_{jk}^{(i)}}\right] \\ &= \sum_{i=1}^{N} \sum_{j=1}^{2} \sum_{k=1}^{2} z_{jk}^{(i)} [\log g_k^{(i)} + \log g_{j|k}^{(i)} + \log f_{jk}(d_i)] \end{aligned} \tag{7.71}$$

给出。用式(7.70)代入式(7.71)且忽略常数 $-(1/2)\log(2\pi)$，因此可以写成

$$L_c(\boldsymbol{\theta}) = \sum_{i=1}^{N} \sum_{j=1}^{2} \sum_{k=1}^{2} z_{jk}^{(i)} \left[\log g_k^{(i)} + \log g_{j|k}^{(i)} - \frac{1}{2}(d_i - y_{jk}^{(i)})^2\right] \tag{7.72}$$

[384]

比较式(7.72)和式(7.65)，通过将指示器变量作为缺失数据加入到可观察的数据集中，立即看出所获得的计算上的好处：最大似然估计问题被解耦为针对单个专家的一组回归问题和针对门网的一组可分离的多项式分类问题。

为了继续应用 EM 算法，通过求完整数据对数似然函数 $L_c(\boldsymbol{\theta})$ 的期望值我们首先启动 E 步，表示为

$$\begin{aligned} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(n)) &= E[L_c(\boldsymbol{\theta})] \\ &= \sum_{i=1}^{N} \sum_{j=1}^{2} \sum_{k=1}^{2} E[z_{jk}^{(i)}] \cdot \left(\log g_k^{(i)} + \log g_{j|k}^{(i)} - \frac{1}{2}(d_i - y_{jk}^{(i)})^2\right) \end{aligned} \tag{7.73}$$

其中针对指示器变量求期望值，因为 $z_{jk}^{(i)}$ 是惟一不可观察的变量。因此，用式(7.68)代入式(7.73)，得到(Jordan and Jacobs, 1994)

$$Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n)) = \sum_{i=1}^{N}\sum_{j=1}^{2}\sum_{k=1}^{2} h_{jk}^{(i)}\left(\log g_k^{(i)} + \log g_{j|k}^{(i)} - \frac{1}{2}(d_i - y_{jk}^{(i)})^2\right) \tag{7.74}$$

该算法的 M 步要求对 $\boldsymbol{\theta}$ 求 $Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))$ 的最大值。参数向量 $\boldsymbol{\theta}$ 由两组突触权值组成：一组属于门网而另一组属于专家。从前面的讨论注意下面的事实：

- 专家的突触权值决定 $y_{jk}^{(i)}$，它也进入 $h_{jk}^{(i)}$ 的定义中。因此专家仅仅通过项 $h_{jk}^{(i)}(d_i - y_{jk}^{(i)})^2$ 影响表达式 $Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))$。
- 门网的突触权值决定概率 $g_k^{(i)}$，$g_{j|k}^{(i)}$ 和 $h_{jk}^{(i)}$。门网仅仅通过项 $h_{jk}^{(i)}(\log g_k^{(i)} + \log g_{j|k}^{(i)})$ 影响表达式 $Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))$ 的。

因此，在一个两层结构的 HME 中算法的 M 步简化为三个最优化问题：

$$\mathbf{w}_{jk}(n+1) = \arg\min_{\mathbf{w}_{jk}} \sum_{i=1}^{N} h_{jk}^{(i)}(d_i - y_{jk}^{(i)})^2 \tag{7.75}$$

$$\mathbf{a}_j(n+1) = \arg\max_{\mathbf{a}_j} \sum_{i=1}^{N}\sum_{k=1}^{2} h_{jk}^{(i)} \log g_k^{(i)} \tag{7.76}$$

385

$$\mathbf{a}_{jk}(n+1) = \arg\max_{\mathbf{a}_{jk}} \sum_{i=1}^{N}\sum_{l=1}^{2} h_l^{(i)} \sum_{m=1}^{2} h_{m|l}^{(i)} \log g_{m|l}^{(i)} \tag{7.77}$$

在式(7.75)至(7.77)的最优化中，h 是固定的；h 虽然是一个参数的函数，但是并不对 h 求导数。另外也要注意这些等式右边的所有量都是指时间步 n 时的取值。

式(7.75)中关于专家的最优化是加权的最小平方估计问题。剩下的式(7.76)和(7.77)关于门网的最优化问题是最大似然估计问题[9]。注意，虽然这些公式只是针对两层结构的，但是它们很容易扩充到任意多层的结构中去。

7.14 小结和讨论

在建模、模式分类和回归问题的研究中，有两个极端情况需要考虑：

1. 简单模型，它提供对感兴趣问题的见解，但缺乏精确度。
2. 复杂模型，该模型提供精确结果但缺乏见解。

单个的模型既简单又精确也许是不可能的。在本章的第二部分，CART 是一个简单模型的例子，该模型用硬决策将输入空间分割成一系列子空间，每个子空间有自己的专家。不幸的是，硬决策的使用带来一些信息的损失，因而带来性能上的损失。在另一个方面，多层感知器(MLP)是用嵌套非线性形式保持训练数据信息的复杂模型。但是，它使用黑盒方法用单个函数整体拟合数据，因而缺乏对问题的见解。HME 模型，代表一种动态类型的委员会机器，是两个极端之间的一种折中模型，有着 MLP 和 CART 的共同特征：

- HME 模型的结构和 CART 类似，但不同之处在于前者是对输入空间的软分割，而后者是硬分割。
- HME 模型类似于 MLP 使用嵌套的非线性形式，但不是为了输入 - 输出映射的目的，而是为了输入空间的分割。

在本章我们强调用于设计 HME 模型的两种工具的使用：

- 在处理模型选择问题的时候，CART 是作为结构基础
- EM 算法是通过迭代计算模型参数的最大似然估计值来解决参数估计问题的。

EM 算法经常能保证似然值向上(uphill)移动。因而，通过使用 7.8 节描述的方式应用

CART去初始化EM算法，可以期望EM算法能产生的泛化性应该比CART算法建立的初始条件产生的泛化性能好。

假如感兴趣的应用是最大似然估计，比如在建模中，EM算法是重要的和基本的。一个有意思的建模应用在Jacobs,Jordan and Barto(1991b)中描述，其中一个ME模型被训练去完成“什么/哪里”任务。在这个任务中，模型被要求去决定目标是什么，目标在可视区域的什么地方。在学习的过程中，应用了两个专家，它们中的每一个是专门承担任务的一个方面。对于一个特定的输入，两个专家都会产生输出。但是，由门网决定对输入适当的混合。Jacobs等人的报告的成功结果表明，决定任务分配的本质可能是基于在任务的要求和模型的计算属性之间的匹配，而不是基于任务本身(Elman et al.,1996)。 386

这个讨论以返回本章第一部分学习过的另外一类委员会机器的研究作为结束。ME和HME模型依赖于使用由输入信号激活的门网来融合被模型中的专家所获得的知识；但是一个基于总体平均或者推举的委员会机器，依赖于学习算法本身去做整合，归纳如下：

1. 总体平均通过对以下两个措施的结合以一种聪明的方式提高它的误差性能：

- 归结为偏置的误差减少，通过有意识地过拟合委员会机器中的单个的专家。
- 归结于方差的误差减少，通过在训练单个专家时使用不同的初始条件，然后总体平均各自的输出。

2. 推举通过本身独特的方法来提高误差性能。在这种情况下，只要求单个专家的性能比随机猜想稍微好一点。专家的弱学习模型被转化成强学习模型，因而该委员会机器的误差可以变得任意小。取得这种非凡的转化是通过某种方式对输入数据的分布进行过滤，使得弱学习模块(即专家)最终学到整个分布，或者如同自举那样，通过根据一定的概率分布对训练样本进行重采样。自举比通过过滤的推举的优越之处在于它的训练例子的数目是固定的。

注释和参考文献

[1] 在Perrone(1993)中讨论*总体平均方法*，其中包括该主题的大量文献。有关这个主题的其他参考文献包括Wolpert(1992)和Hashem(1997)。

[2] 几个神经网络先驱者建议使用不同初始条件的总体平均设计委员会机器。但是，在Naftaly et al.(1997)中给出的统计分析以及那里描述的由初始条件空间的总体平均设计训练委员会机器的过程看来是其中第一次。在那篇文章中，基于太阳黑子数据和能量－预测竞争数据得出实验结果。在两种情况下对初始条件空间求平均值显示方差显著下降。

根据Naftaly et al.(1997)，在用初始条件空间的总体平均设计委员会机器时不提倡使用流行的诸如权值衰减和早期停止等训练约束条件。

[3] 推举理论的主要参考文献和相关的实验研究以时间为序或前或后可排序如下：Schapire(1990)，Drucker et al.(1993,1994)，Freund(1995)，Breiman(1996b)，Freund and Schapire(1996a,1996b,1997)，Schapire(1997)和Schapire et al.(1997)。关于推举的三个基本方法的首批参考文献分别如下： 387

- 滤波：Schapire(1990)
- 重新采样：Freund and Schapire(1996a)
- 重新加权：Freund(1995)

[4] Jacobs，Jordan，Nowlan 和 Hinton 在他们 1991a 的文章中首次讨论利用混合专家实现复杂映射函数的思想。这个模型的发展归功于(1)Nowlan(1990)提出的一个建议：将非监督学习的竞争自适应看作试图使简单概率分布的混合拟合一组数据，(2)在 Jacobs(1990)的博士学位论文中利用相似的组件结构和不同的代价函数所发展的思想。

[5] 最大似然估计器有一些希望的性质。在相当一般条件下可以证明下列渐进性质(Kmenta,1971)：

(i)最大似然估计器是相容的。令 $L(\boldsymbol{\theta})$ 表示对数似然函数，θ_i 为参数向量 $\boldsymbol{\theta}$ 的分量：偏导数 $\partial L/\partial\theta_i$ 称为分值。我们说一个最大似然函数估计器是相容的指的是使得分值 $\partial L/\partial\theta_i$ 等于 0 时 θ_i 的取值随估计中样本趋于无穷而依概率收敛到 θ_i 的真实值。

(ii)最大似然估计器是渐进有效的。也就是

$$\lim_{N\to\infty}\left[\frac{\mathrm{var}[\theta_i-\hat{\theta}_i]}{I_{ii}}\right]=1 \qquad 对所有\ i$$

其中 N 为样本数目，$\hat{\theta}_i$ 为 θ_i 的最大似然估计，而且 I_{ii} 为 Fisher 信息矩阵的逆矩阵的第 i 个对角元素。Fisher 信息矩阵定义为

$$J=-\begin{bmatrix} E\left[\frac{\partial^2 L}{\partial\theta_1^2}\right] & E\left[\frac{\partial^2 L}{\partial\theta_1\partial\theta_2}\right] & \cdots & E\left[\frac{\partial^2 L}{\partial\theta_1\partial\theta_M}\right] \\ E\left[\frac{\partial^2 L}{\partial\theta_2\partial\theta_1}\right] & E\left[\frac{\partial^2 L}{\partial\theta_2^2}\right] & \cdots & E\left[\frac{\partial^2 L}{\partial\theta_2\partial\theta_M}\right] \\ \vdots & \vdots & & \vdots \\ E\left[\frac{\partial^2 L}{\partial\theta_M\partial\theta_1}\right] & E\left[\frac{\partial^2 L}{\partial\theta_M\partial\theta_2}\right] & \cdots & E\left[\frac{\partial^2 L}{\partial\theta_M^2}\right] \end{bmatrix}$$

其中 M 为参数向量 $\boldsymbol{\theta}$ 的维数。

(iii)最大似然函数估计器是渐进 Gauss 的。也就是，当样本数趋于无穷时，最大似然估计 $\hat{\boldsymbol{\theta}}$ 的每一个元素为 Gauss 分布。

实际上，我们发现最大似然函数估计器的大样本(渐进)性质对样本数 $N\geq 50$ 就保持得相当好。

[6] Newcomb(1886)的文章考虑两个单变元 Gauss 分布的混合参数估计，看起来这是文献报告中最早的一个 EM 类型过程的参考文章：

"EM 算法"的名称由 Dempster，Laird 和 Rubin 在他们 1977 奠基性的文章中创造的。在那篇文章中第一次给出不同推广层次下从不完整数据中计算最大似然估计的 EM 算法的公式。

Mclachlan and Krishnan(1997)以书的形式第一次统一考虑 EM 算法的理论、方法和应用，它的历史以及推广。

[7] 在相当一般条件下 EM 算法计算的似然值收敛到稳定值。Wu(1983)给出 EM 算法收敛性质的详细考虑。但是 EM 算法并不总是导致似然函数的局部或全局最大值。在 McLachlan and Krishnan(1997)撰写的书的第 3 章，给出两个不收敛的例子，在一个例子中算法收敛到鞍点，而在另一个例子中算法收敛到似然函数的局部最小值。

388

[8] 利用参数向量的先验信息，EM 算法也可以处理 Bayes 最大后验(maximum a posterior, MAP)估计；参看习题 7.11。利用 Bayes 规则，对于给定一组观察 $\mathbf{x}$ 可以把参数向量 $\boldsymbol{\theta}$

的条件密度函数表示为

$$f_{\Theta}(\boldsymbol{\theta} \mid \mathbf{x}) = \frac{f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\theta}) f_{\Theta}(\boldsymbol{\theta})}{f_{\mathbf{X}}(\mathbf{x})}$$

由这个关系，我们能够看出最大化后验密度 $f_{\Theta}(\boldsymbol{\theta}|\mathbf{x})$等价于最大化积函数 $f_{\mathbf{X}}(\mathbf{x}|\boldsymbol{\theta})f_{\Theta}(\boldsymbol{\theta})$，因为 $f_{\mathbf{X}}(\mathbf{x})$是独立于 $\boldsymbol{\theta}$ 的。概率密度函数 $f_{\Theta}(\boldsymbol{\theta})$表示 $\boldsymbol{\theta}$ 的可用先验信息。给定 $\mathbf{x}$ 之后最大化概率密度函数 $f_{\Theta}(\boldsymbol{\theta}|\mathbf{x})$提供参数向量 $\boldsymbol{\theta}$ 的最可能估计。在这种估计的背景下有两点值得注意：

- 对 $\boldsymbol{\theta}$ 极大化 $f_{\mathbf{X}}(\mathbf{x}|\boldsymbol{\theta})$表示最大似然估计，是最大后验估计的简化形式，简化的意思是不用先验信息。
- 使用先验信息与正则化是同步的，这(回忆第 5 章)相当于光滑的输入 – 输出映射。

Waterhouse et al.(1996)给出混合专家模型用于估计参数的 Bayes 框架，那里描述的 Bayes 方法克服了著名的“过拟合”现象，当用最大似然函数推断时“过拟合”导致具有高方差的估计。

[9] 在式(7.76)和(7.77)中描述的最大似然估计问题可用一个有效算法，称为*迭代重新加权最小二乘*(iteratively reweighted least-squares, IRLS)*算法*；关于 IRLS 算法的描述可参看 McCullagh and Nelder(1989)。

习题

总体平均

7.1 考虑由 K 个专家组成的委员会机器。第 k 个专家的输入 – 输出函数表示为 $F_k(\mathbf{x})$，其中 $\mathbf{x}$ 为输入向量，$k = 1,2,\cdots,K$。每个专家各自输出的线性组合形成总的输出，定义为

$$y = \sum_{k=1}^{K} w_k F_k(\mathbf{x})$$

其中 w_k 是赋值给 $F_k(\mathbf{x})$的线性权值。要求估计 w_k 的值使得 y 提供了相应于 $\mathbf{x}$ 的期望输出 d 的最小平方估计。给定训练数据集 $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$，求 w_k 的所需值解决这个参数估计问题。

推举

7.2 比较通过过滤的推举和自举在计算上的优缺点。

7.3 通常，推举在弱学习模型(即具有相对低泛化误差率的学习模型)表现最好。但是，假设给你一个强学习模型，即具有高泛化误差率的学习模型。若你处理大小固定的训练样本，这时怎样通过过滤推举和自举处理这种情况？ 389

混合专家

7.4 考虑分段线性任务，描述为

$$F(x_1, x_2, \cdots, x_{10}) = \begin{cases} 3x_2 + 2x_3 + x_4 + 3 + \varepsilon & 若\ x_1 = 1 \\ 3x_5 + 2x_6 + x_7 - 3 + \varepsilon & 若\ x_1 = -1 \end{cases}$$

为了比较，利用下列网络配置：

1. 多层感知器：“10→10→1”网络
2. 混合专家： 门网：10→2

专家网络：10→1

比较这两个网络的计算复杂性。

7.5 式(7.30)的条件概率密度函数描述的ME模型是基于标量回归模型，其中误差是具有零均值单位方差的Gauss分布。

(a)对于对应于多重回归模型的ME模型的更一般情况，重新构造这个等式的公式，其中期望响应是具有多维数 q 的向量，而误差是具有零均值和协方差矩阵为 $\mathbf{\Sigma}$ 的多元Gauss分布。

(b)这个重新构造公式的ME模型和图7-8所示的ME模型如何不同?

7.6 推导用于训练混合专家模型的随机梯度算法。

分层混合专家

7.7 (a)构造具有三层的HME模型的框图，假设模型利用二叉决策树。

(b)对(a)中描述的HME模型的非终端节点写出后验概率。说明在求这些概率值所涉及的计算的递归性。

(c)对(a)中描述的HME模型，构造条件概率密度函数的公式。

7.8 讨论HME模型和径向基函数(RBF)网络的相似之处和不同之处。

7.9 对于具有两层的HME模型的训练，推导描述它的随机梯度算法的方程。假设该模型应用二叉决策树。

EM算法和它在HME模型中的应用

7.10 证明在式(7.62)中描述的EM算法的单调上升性质。为了这个推导，做下面的事：

(a)令

390

$$k(\mathbf{r} \mid d,\boldsymbol{\theta}) = \frac{f_c(\mathbf{r} \mid \boldsymbol{\theta})}{f_D(d \mid \boldsymbol{\theta})}$$

代表给定观察 d 和参数向量 $\boldsymbol{\theta}$ 时扩充后的完全数据向量 $\mathbf{r}$ 的条件概率密度函数，因而不完整数据对数似然函数可表示为

$$L(\boldsymbol{\theta}) = L_c(\boldsymbol{\theta}) - \log k(\mathbf{r} \mid d,\boldsymbol{\theta})$$

其中 $L_c(\boldsymbol{\theta}) = \log f_c(\mathbf{r} \mid \boldsymbol{\theta})$ 为完全数据的对数似然函数。给定 d，对 $\mathbf{r}$ 的条件分布取 $L(\boldsymbol{\theta})$ 的期望值，证明

$$L(\boldsymbol{\theta}) = Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n)) - K(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))$$

其中

$$K(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n)) = E[\log k(\mathbf{r} \mid d,\hat{\boldsymbol{\theta}})]$$

因而证明

$$L(\hat{\boldsymbol{\theta}}(n+1)) - L(\hat{\boldsymbol{\theta}}(n)) = [Q(\hat{\boldsymbol{\theta}}(n+1),\hat{\boldsymbol{\theta}}(n)) - Q(\hat{\boldsymbol{\theta}}(n),\hat{\boldsymbol{\theta}}(n))] - [K(\hat{\boldsymbol{\theta}}(n+1),\hat{\boldsymbol{\theta}}(n)) - K(\hat{\boldsymbol{\theta}}(n),\hat{\boldsymbol{\theta}}(n))]$$

(b)若 $g(\cdot)$ 为凸函数，u 为随机变量，Jensen不等式可陈述为

$$E[g(u)] \geqslant g(E[u])$$

其中 E 为期望算子；而且若 $g(\cdot)$ 为严格凸，那么等式成立意味着以概率1有 $u = E[u]$(Cover and Thomas,1991)。利用Jensen不等式证明

$$K(\hat{\boldsymbol{\theta}}(n+1),\hat{\boldsymbol{\theta}}(n)) - K(\hat{\boldsymbol{\theta}}(n),\hat{\boldsymbol{\theta}}(n)) \leqslant 0$$

从而证明式(7.62)对 $n = 1,2,\cdots$ 成立。

7.11 EM算法很容易修改，使之适用于参数向量 $\boldsymbol{\theta}$ 的最大后验(MAP)估计。利用Bayes

规则，修改 EM 算法 E－步和 M－步提供这个估计。

7.12 对于给定任务，用 EM 算法训练 HME 和利用反向传播算法训练 MLP 达到相似水平的性能，直观上我们希望 HME 超过 MLP 的计算复杂性。给出支持或反对这个陈述合理性的论证。

7.13 判断式(7.66)至(7.68)描述的指示器变量和相应的后验概率关系的合理性。

7.14 假设期望响应 d 是标量，式(7.75)描述图 7-11 中 HME 模型的专家网络的加权最小平方最优化。当期望响应为多维时怎样修改这种关系？ 391

第8章 主分量分析

8.1 简介

神经网络的一个重要特征就是它们具有向环境学习并通过学习改善本身性能的能力。在前面四章中主要讨论了监督学习的算法，外部教师为它们提供一组有意义的目标。目标采取期望输入－输出映射的形式，要求网络对这个映射进行逼近。在本章和后面的三章中，我们学习*自组织学习*(self-organized learning)或*无监督学习*(unsupervised learning)。自组织学习的目的是为了发现输入数据中的重要模式和特征，而这些发现是无教师的。为了这样做，系统需要提供一组局部性的规则，这些规则能使它学会计算具有特殊期望性质的输入－输出映射。术语“局部”意味突触权值的改变只与邻近单元的状态有关。用于自组织学习的神经网络结构模型比用于监督学习的模型更接近生物神经系统模型。这并不奇怪，因为网络组织的过程是脑组织过程的基础。

自组织结构有各种各样的形式。例如，它可能由一个*输入(源)层*和*输出(表示)层*组成，输入层到输出层之间有前馈连接，输出层各单元间有侧向连接。另一个例子是前馈网络，由多层组成，其中自组织是以层到层为基础进行的。在上述两个例子中，学习过程都是按照预定的规则和对输入(激活)模式的响应重复修改系统中的所有突触权值，直到形成一种最终设置。

本章只讨论基于 Hebb 学习的自组织系统，主要集中于*主分量分析*(principal components analysis)，这是统计模式识别和信号处理中进行数据压缩通用的一种标准方法。 392

本章的组织

本章的材料组织如下。在 8.2 节用定性论据描述自组织系统的基本原理。随后在 8.3 节中介绍主分量分析，这也是本章其余部分讨论的自组织系统的基础。

在掌握基本背景材料后，接下来学习一些具体的自组织系统。8.4 节描述由单个神经元组成的简单系统，它以自组织方式抽出第一个主分量。8.5 节将讨论更复杂的系统，它为具有前馈连接的单层网络形式，通过对以前简单系统的扩展，抽出所有的主分量。在 8.6 节将给出一个关于图像编码的具体实例演示这个过程。8.7 节将阐述另一个具有相似功能的自组织系统，这个系统更加复杂，因为它包含侧向连接。

在 8.8 节中给出利用神经网络进行主分量分析的各种算法的分类。随后 8.9 节在数据分类的基础上将算法分成自适应方法和集中式方法。

在 8.10 节描述主分量分析基于内积核思想的非线性形式，内积核按照第 6 章的支持向量机模型中讨论的 Mercer 定理定义。

在 8.11 节以对主分量分析的一些最后思考结束本章。

8.2 自组织的一些直观原则

像前面提到的那样，自组织(无监督)学习按照预定的规则和对激活模式的响应重复修改

神经网络的突触权值，直到形成一种最终设置。当然，问题的关键是，怎样从自组织中形成一个有用的设置。答案本质上来自于下面的观察(Turing,1952)：

局部相互作用可以导致整体的序。

这个观察具有重要意义；它适用于脑和人工神经网络。尤其，网络相邻神经元之间许多最初随机的局部作用，能够结合成整体有序的状态，并最终在空间模式或时间节奏上形成连贯行为；这些是自组织的本质。

网络组织在两个不同层次的发生，两个层次之间以反馈环的形式相互作用。这两个层次为：

- *活动性*。由给定网络对输入信号的响应产生某种活动模式。
- *连接性*。由于突触可塑性，网络连接强度(突触权值)由于响应活动模式中的神经信号得以修改。

393

为了达到网络的自组织(而非稳定)，在突触权值变化和活动模式变化之间的反馈必须是正的。因此，可以得到自组织系统的第一个基本原则(von der Malsburg,1990a)：

原则1 突触权值的修改趋向于自增强。

突触权值的修改必须基于局部可用信号，即前突触和后突触的信号，自增强过程被这种要求所限制。自增强和局部性的要求确定这样的机制，强的突触导致前突触信号和后突触信号相一致。通过这种一致性又使突触的强度增加。这里所描述的机制实际上是Hebb学习假设的重述。

为了使系统稳定，必须存在对“有限”资源(例如输入的数量和能量资源)的一些竞争形式。具体地，网络中的一些突触强度增加必须以其他突触的减弱来补偿。因此，只有“成功”的突触才能生长，而不成功的将减弱并最终消失。从这个观察结果可得到自组织的第二个原则(von der Malsburg,1990a)：

原则2 资源的有限导致突触间竞争，从而导致牺牲其他突触来选择最活跃(即最适合)的生长突触。

突触的可塑性也使这一原则成为可能。

对下一步的观察，我们注意单个突触不能有效地产生满意的结果。为了达到上述效果，需要一组突触间的协作，而这些突触聚集于一个特定的神经元且带有足够强大的相同信号以激活该神经元。因此我们可以抽象出第三个自组织原则(von der Malsburg,1990a)：

原则3 突触权值的修改趋向于协作。

尽管网络中存在竞争，活跃突触的出现能够增强其他突触的适应。这种协作形式的出现可能归因于突触的可塑性，或归因于外部环境中出现适宜的条件同时刺激前突触神经元。

上面所描述的三个自组织原则只与网络本身有关。然而为了自组织学习执行有用的信息处理功能，环境提供给网络的激活模式中必须存在冗余(redundancy)。冗余问题将在第10章Shannon信息论框架中讨论。现在足以提出自组织学习的最后一个原则如下(Barlow,1989)：

原则4 激活模式中次序和结构表示冗余信息，神经网络以知识的形式得到这些冗余信息，这是自组织学习的必要前提。

394

我们可以从统计参数的观测中获得这些知识，例如，从输入数据的均值、方差和相关矩阵。

关于自组织学习的原则 1 至原则 4 为本章讨论主分量分析和下一章描述 Kohonen 自组织映射的自适应算法提供神经生物学的基础。这些原则在其他许多受神经生物学考虑激励的自组织模型中也被采用。值得一提的这样一种模型是哺乳动物视觉系统的 Linsker 模型(Linsker,1986)。

自组织的特征分析

视觉系统中的信息处理是分阶段的。具体地，一些简单的特征如对比度和边缘方向是在系统的早期阶段分析的，而更精致复杂的特征则在后期阶段进行分析。图 8-1 表示与视觉系统相似的模型网络的整体结构。在 Linsker 的模型中，图 8-1 的网络神经元组织成二维层，从一层到下一层具有局部前馈连接。每个神经元只接受前一层位于一个覆盖区内有限数目神经元的信息，此区域称为接受域(receptive field)。网络接受域在突触的形成过程中起关键作用，因为它们使一层中的神经元对前一层神经活动的空间相关性的反应成为可能。假设下面两个结构特征：

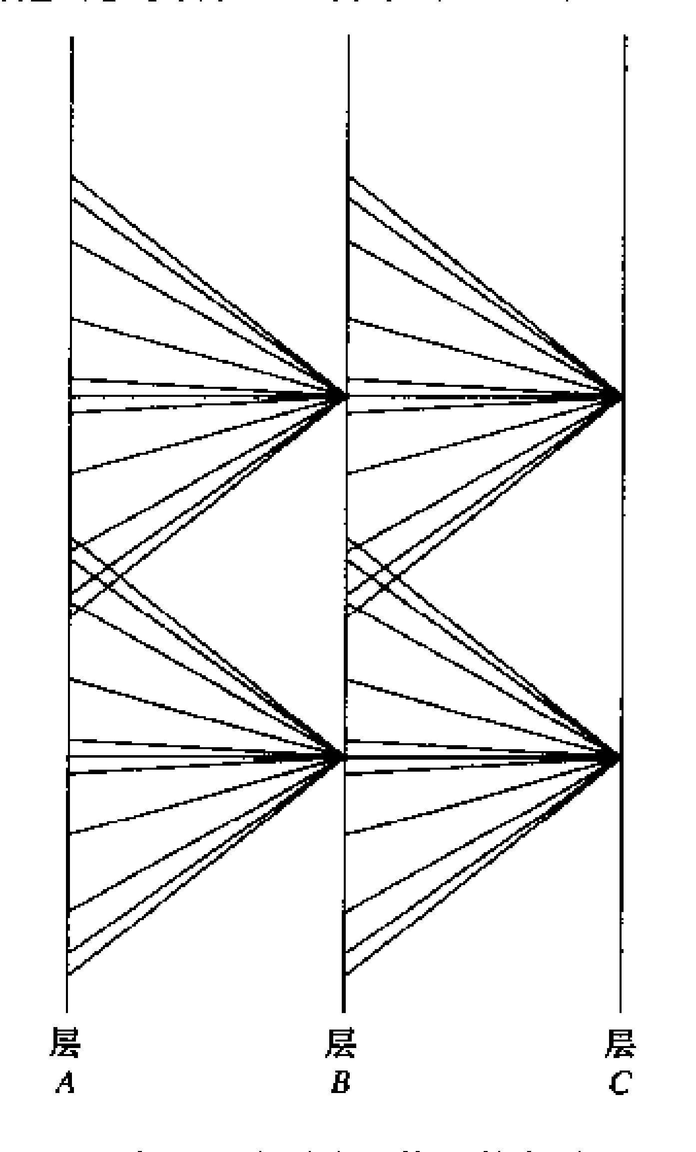


图 8-1 自适应网络组件布局

1. 在整个神经元形成过程中，一旦突触连接被选择，其位置就固定了。

2. 每个神经元都是一个线性组合器。

模型结合 Hebb 型突触修改的协作和竞争学习的方面使得网络输出最优区分输入总体，这需要通过自组织学习从一层到一层的基础上处理。即学习过程在处理下一层之前允许全面形成该层自身的自组织特征 - 分析(feature-analyzing)特性。在 Linsker(1986)中模拟结果与猫和猴子的视觉形成的早期具有非常相似的性质。认识到视觉系统的高度复杂性，而 Linsker 考虑的非常简单的模型能形成相似的特征 - 分析神经元，这的确值得注意。此点并非意味着哺乳动物的视觉系统的特征 - 分析神经元形成的方式与上面的 Linsker 模型描述的方式完全相同。相反，它只能说明按照 Hebb 学习规则形成突触权值，再由这种相对简单的层状网络就可产生这种结构。

395

但是，在本章中我们主要的兴趣是主分量分析和利用基于 Hebb 学习的自组织系统怎样实现它。

8.3 主分量分析

在统计模式识别中，一个常见的问题就是特征选择或特征提取。特征选择是指将数据空间变换到特征空间的过程，在理论上与原始数据空间具有相同的维数。然而，我们希望设计一种变换使得数据集由维数较少的“有效”特征来表示，而不减少原始数据所包含的内在信息内容；换句话说，数据集进行了*维数压缩*。具体来说，假设有一个 m 维的向量 $\mathbf{x}$，希望压缩到 l 维，其中 $l < m$。如果我们简单截断 $\mathbf{x}$，所带来的均方误差等于舍掉的各分量的方差之和。因此提出下面的问题：是否存在一个可逆的线性变换 $\mathbf{T}$，使得对 $\mathbf{Tx}$ 的截断在均方误差意义下最优？显然要求变换后的某些分量具有较低的方差。*主分量分析*(principal components

analysis，在通信理论中也叫 Karhunen-Loève 变换)能最大程度地减少方差，并因而是正确的选择。在本章我们讨论基于 Hebb 学习算法来完成数据向量的主分量分析[1]。

令 $\mathbf{X}$ 为表示环境的 m 维随机向量。假设 $\mathbf{X}$ 均值为零，即

396

$$E[\mathbf{X}] = \mathbf{0}$$

其中 E 是统计学习中的期望运算符。如果 $\mathbf{X}$ 的均值不是 0，在执行分析之前先减去其均值。令 $\mathbf{q}$ 表示 m 维单位向量，$\mathbf{X}$ 在其上投影。这个投影被定义为向量 $\mathbf{X}$ 和 $\mathbf{q}$ 的内积，表示为

$$A = \mathbf{X}^T\mathbf{q} = \mathbf{q}^T\mathbf{X} \tag{8.1}$$

满足约束条件

$$\|\mathbf{q}\| = (\mathbf{q}^T\mathbf{q})^{1/2} = 1 \tag{8.2}$$

投影 A 也是随机变量，其均值和方差与 $\mathbf{X}$ 的统计有关。由假设 $\mathbf{X}$ 的均值为 0，推知 A 的均值也为 0：

$$E[A] = \mathbf{q}^TE[\mathbf{X}] = 0$$

A 的方差与其均方值相同，可写为

$$\sigma^2 = E[A^2] = E[(\mathbf{q}^T\mathbf{X})(\mathbf{X}^T\mathbf{q})] = \mathbf{q}^TE[\mathbf{X}\mathbf{X}^T]\mathbf{q} = \mathbf{q}^T\mathbf{R}\mathbf{q} \tag{8.3}$$

$m\times m$ 矩阵 $\mathbf{R}$ 是随机向量 $\mathbf{X}$ 的自相关矩阵，正式定义为向量 $\mathbf{X}$ 和它自己的外积的期望，表示为

$$\mathbf{R} = E[\mathbf{X}\mathbf{X}^T] \tag{8.4}$$

我们观察到相关矩阵 $\mathbf{R}$ 是对称的，即

$$\mathbf{R}^T = \mathbf{R} \tag{8.5}$$

由这个性质知，如果 $\mathbf{a}$ 和 $\mathbf{b}$ 为任意 $m\times 1$ 向量，那么

$$\mathbf{a}^T\mathbf{R}\mathbf{b} = \mathbf{b}^T\mathbf{R}\mathbf{a} \tag{8.6}$$

由式(8.3)看出，投影 A 的方差 σ^2 是单位向量 $\mathbf{q}$ 的函数，可以写为

$$\psi(\mathbf{q}) = \sigma^2 = \mathbf{q}^T\mathbf{R}\mathbf{q} \tag{8.7}$$

基于此我们可以认为 $\psi(\mathbf{q})$为方差探针(variance probe)。

主分量分析的特征结构

下面讨论的问题是在欧几里德范数的约束条件下，找出单位向量 $\mathbf{q}$ 沿 $\psi(\mathbf{q})$所具有的极(extremal)值或稳定(stationary)值(局部最大或最小)。这个问题的解决依赖于输入向量的相关矩阵 $\mathbf{R}$ 的特征结构。如果 $\mathbf{q}$ 为单位向量使得方差探针 $\psi(\mathbf{q})$具有极值，那么对单位向量 $\mathbf{q}$ 任意小的扰动 $\delta\mathbf{q}$，我们发现直到 $\delta\mathbf{q}$ 的一阶项将有

397

$$\psi(\mathbf{q}+\delta\mathbf{q}) = \psi(\mathbf{q}) \tag{8.8}$$

现在，从式(8.7)给出的方差探针定义，我们有

$$\psi(\mathbf{q}+\delta\mathbf{q}) = (\mathbf{q}+\delta\mathbf{q})^T\mathbf{R}(\mathbf{q}+\delta\mathbf{q}) = \mathbf{q}^T\mathbf{R}\mathbf{q} + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} + (\delta\mathbf{q})^T\mathbf{R}\delta\mathbf{q}$$

在第 2 个等式中，已经利用式(8.6)。忽略项$(\delta\mathbf{q})^T\mathbf{R}\delta\mathbf{q}$ 并利用式(8.7)的定义，可以写成

$$\psi(\mathbf{q}+\delta\mathbf{q}) = \mathbf{q}^T\mathbf{R}\mathbf{q} + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} = \psi(\mathbf{q}) + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} \tag{8.9}$$

因此将式(8.8)代入式(8.9)得

$$(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} = 0 \tag{8.10}$$

对 $\mathbf{q}$ 而言，任意扰动 $\delta\mathbf{q}$ 是不允许的；相反对扰动进行限制，仅使 $\mathbf{q}+\delta\mathbf{q}$ 的欧几里德范数为 1 的扰动是允许的，即

$$\| \mathbf{q} + \delta\mathbf{q} \| = 1$$

或等价地

$$(\mathbf{q} + \delta\mathbf{q})^T(\mathbf{q} + \delta\mathbf{q}) = 1$$

因此，根据式(8.2)，我们要求对 $\delta\mathbf{q}$ 的一阶项有

$$(\delta\mathbf{q})^T\mathbf{q} = 0 \tag{8.11}$$

这意味着，扰动 $\delta\mathbf{q}$ 必须与 $\mathbf{q}$ 正交，因此仅在 $\mathbf{q}$ 的垂直方向上变化是允许的。

通常单位向量 $\mathbf{q}$ 在物理意义上是无量纲的。从而如果结合式(8.10)和(8.11)，那么我们必须在式(8.11)中引入一个比例因子 λ 使得它和相关矩阵 $\mathbf{R}$ 中的元素有相同的量纲。于是可以写成

$$(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} - \lambda(\delta\mathbf{q})^T\mathbf{q} = 0$$

或等价地

$$(\delta\mathbf{q})^T(\mathbf{R}\mathbf{q} - \lambda\mathbf{q}) = 0 \tag{8.12}$$

式(8.12)成立的充要条件为

$$\mathbf{R}\mathbf{q} = \lambda\mathbf{q} \tag{8.13}$$

398

这个方程控制单位向量 $\mathbf{q}$ 使得方差探测值 $\psi(\mathbf{q})$ 有极值。

式(8.13)被认为是特征值问题，通常在线性代数中碰到(Strang, 1980)。仅对特殊的 λ 值问题有非平凡解(即 $\mathbf{q} \neq \mathbf{0}$)，$\lambda$ 被称为相关矩阵 $\mathbf{R}$ 的特征值，对应的 $\mathbf{q}$ 被称为特征向量。相关矩阵的特征值必须是非负数。假设它的特征值互不相同，则对应的特征向量是惟一的。令 $m \times m$ 矩阵 $\mathbf{R}$ 的特征值为$\lambda_1, \lambda_2, \cdots, \lambda_m$，对应的特征向量分别是$\mathbf{q}_1, \mathbf{q}_1, \cdots, \mathbf{q}_m$。我们可写成

$$\mathbf{R}\mathbf{q}_j = \lambda_j\mathbf{q}_j, \qquad j = 1, 2, \cdots, m \tag{8.14}$$

令相应特征值按降序排列，即

$$\lambda_1 > \lambda_2 > \cdots > \lambda_j > \cdots > \lambda_m \tag{8.15}$$

这样 $\lambda_1 = \lambda_{max}$。令对应的特征向量用于构成一个 $m \times m$ 矩阵

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_j, \cdots, \mathbf{q}_m] \tag{8.16}$$

我们可以结合式(8.14)中的 m 个方程为一个方程组：

$$\mathbf{R}\mathbf{Q} = \mathbf{Q}\mathbf{\Lambda} \tag{8.17}$$

其中 $\mathbf{\Lambda}$ 为 $\mathbf{R}$ 的特征值构成的对角矩阵，即

$$\mathbf{\Lambda} = \mathrm{diag}[\lambda_1, \lambda_2, \cdots, \lambda_j, \cdots, \lambda_m] \tag{8.18}$$

矩阵 $\mathbf{Q}$ 是正交(酉)矩阵，意味着它的列向量(即 $\mathbf{R}$ 的特征向量)满足正性交条件：

$$\mathbf{q}_i^T\mathbf{q}_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases} \tag{8.19}$$

式(8.19)要求不同的特征值。等价地，可写成

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

由此可以推导出矩阵 $\mathbf{Q}$ 的逆矩阵与它的转置矩阵相同，表示为

$$\mathbf{Q}^T = \mathbf{Q}^{-1} \tag{8.20}$$

这意味着可以重写(8.17)为众所周知的正交相似变换形式

$$\mathbf{Q}^T\mathbf{R}\mathbf{Q} = \mathbf{\Lambda} \tag{8.21}$$

或展开为

$$\mathbf{q}_j^T \mathbf{R} \mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \tag{8.22}$$

式(8.21)的正交相似(酉)变换将相关矩阵 $\mathbf{R}$ 变成特征值对角阵。相关矩阵 $\mathbf{R}$ 可以用特征值和特征向量表示为

$$\mathbf{R} = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^T \tag{8.23}$$

399 这称为*谱定理*。对所有 i，外积 $\mathbf{q}_i \mathbf{q}_i^T$ 的秩为 1。

式(8.21)和(8.23)是相关矩阵 $\mathbf{R}$ 的*特征分解*(eigendecomposition)的两个等价表示。

主分量分析和矩阵 $\mathbf{R}$ 的特征分解从根本上来说是一致的，只是从不同的方面观察问题。从式(8.7)和(8.23)可以看出方差探针和特征值的确相等，表示为

$$\psi(\mathbf{q}_j) = \lambda_j, \qquad j = 1,2,\cdots,m \tag{8.24}$$

现在，从主分量分析的特征结构中我们可以概括两个重要发现：

1. 零均值的随机向量 $\mathbf{X}$ 的相关矩阵 $\mathbf{R}$ 的特征向量定义为单位向量 $\mathbf{q}_j$，代表主方向，沿着它们方差探针 $\psi(\mathbf{q}_j)$ 取得极值。

2. 相应的特征值定义方差探针 $\psi(\mathbf{u}_j)$ 的极值。

基本数据表示

令*数据向量* $\mathbf{x}$ 为随机向量 $\mathbf{X}$ 的实现。

由于单位向量 $\mathbf{q}$ 有 m 个可能的解，我们发现数据向量 $\mathbf{x}$ 有 m 个可能的投影需要考虑。特别地，从式(8.1)我们注意

$$a_j = \mathbf{q}_j^T \mathbf{x} = \mathbf{x}^T \mathbf{q}_j, \qquad j = 1,2,\cdots,m \tag{8.25}$$

其中 a_j 是 $\mathbf{x}$ 在单位向量 $\mathbf{u}_j$ 所表示的主方向上的投影。a_j 称作*主分量*，和 $\mathbf{x}$ 具有相同的物理量纲。式(8.25)的公式被看作是一个分析。

为了从投影 a_j 中准确重建原始数据向量 $\mathbf{x}$，我们可以采取下面的步骤。首先，将一组投影 $\{a_j | j = 1,2,\cdots,m\}$ 组合成一个单一的向量，表示为

$$\mathbf{a} = [a_1, a_2, \cdots, a_m]^T = [\mathbf{x}^T\mathbf{q}_1, \mathbf{x}^T\mathbf{q}_2, \cdots, \mathbf{x}^T\mathbf{q}_m]^T = \mathbf{Q}^T\mathbf{x} \tag{8.26}$$

接着我们在式(8.26)的两边左乘矩阵 $\mathbf{Q}$，再利用式(8.20)的关系。因此，原始数据向量 $\mathbf{x}$ 可重建如为

400

$$\mathbf{x} = \mathbf{Q}\mathbf{a} = \sum_{j=1}^{m} a_j \mathbf{q}_j \tag{8.27}$$

它可被看*合成*公式。在这种意义上，单位向量 $\mathbf{q}_j$ 表示数据空间一组基。确实，式(8.27)只是一个坐标变换，根据该变换数据空间中的点 $\mathbf{x}$ 变换到特征空间的点 $\mathbf{a}$。

维数减缩

从统计模式识别的观点看，主分量分析的实际价值在于它为*维数减缩*提供有效的方法。具体地，通过丢弃式(8.27)中方差小的项，保留方差大的项，可以减少有效数据表示所需的特征的数量。令 $\lambda_1, \lambda_2, \cdots, \lambda_l$ 表示相关矩阵 $\mathbf{R}$ 的前 l 个最大特征值。我们*截断*式(8.27)中的 l 项后面的展开式可以得到数据向量 $\mathbf{x}$ 的近似

$$\hat{\mathbf{x}} = \sum_{j=1}^{l} a_j \mathbf{q}_j = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_l] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix}, l \leqslant m \tag{8.28}$$

对给定的原始数据向量 $\mathbf{x}$，可以用式(8.25)计算得到保留在式(8.28)中的主分量如下：

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_l^T \end{bmatrix} \mathbf{x}, \qquad l \leqslant m \tag{8.29}$$

从 $\mathbb{R}^m$ 到 $\mathbb{R}^l$ 的线性投影(即从数据空间到特征空间的映射)是对数据向量 $\mathbf{x}$ 近似表示的*编码器*，如图 8-2a 所示。相应地，从 $\mathbb{R}^l$ 到 $\mathbb{R}^m$ 的线性投影(即特征空间到数据空间的映射)表示为对原始数据向量 $\mathbf{x}$ 近似重构的*解码器*，如图 8-2b 所示。注意式(8.28)、(8.29)中描述的优势(即最大)特征值$\lambda_1, \lambda_2, \cdots, \lambda_l$ 并不参加计算，它们只是分别决定编码器和解码器所使用的主分量的数量。

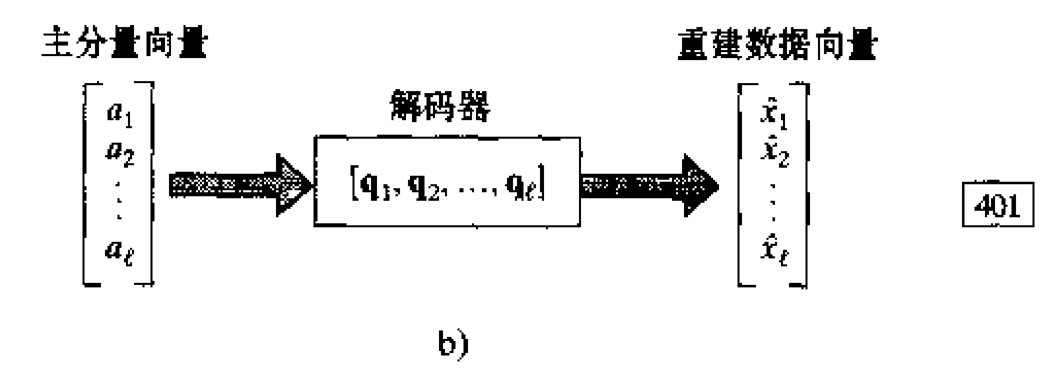


图 8-2 主分量分析的两阶段说明
a)编码 b)解码

逼近误差向量 $\mathbf{e}$ 等于原始数据向量 $\mathbf{x}$ 和逼近数据向量 $\hat{\mathbf{x}}$ 的差，即

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}} \tag{8.30}$$

将式(8.27)和(8.28)代入式(8.30)得

$$\mathbf{e} = \sum_{j=l+1}^{m} a_j \mathbf{q}_j \tag{8.31}$$

误差向量 $\mathbf{e}$ 和逼近数据向量 $\hat{\mathbf{x}}$ 是正交的，如图 8-3 所示。换句话说，$\hat{\mathbf{x}}$ 和 $\mathbf{e}$ 的内积为零。利用式(8.28)和(8.31)这个性质可以表示如下：

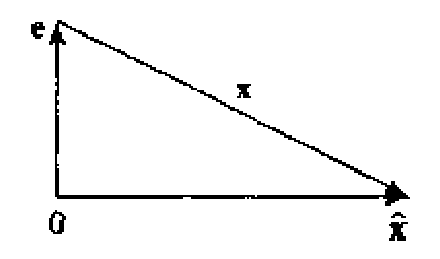


图 8-3 向量 $\mathbf{x}$、它的重建形式 $\hat{\mathbf{x}}$ 和误差向量 $\mathbf{e}$ 的关系示例

$$\begin{aligned} \mathbf{e}^T \hat{\mathbf{x}} &= \sum_{i=l+1}^{m} a_i \mathbf{q}_i^T \sum_{j=1}^{l} a_j \mathbf{q}_j \\ &= \sum_{i=l+1}^{m} \sum_{j=1}^{l} a_i a_j \mathbf{q}_i^T \mathbf{q}_j = 0 \end{aligned} \tag{8.32}$$

其中我们利用了式(8.19)的第二个条件。式(8.32)称作*正交性原理*。

由式(8.7)和(8.22)的第一行，数据向量 $\mathbf{x}$ 的 m 个分量的总方差为

$$\sum_{j=1}^{m} \sigma_j^2 = \sum_{j=1}^{m} \lambda_j \tag{8.33}$$

其中 σ_j^2 是第 j 个主分量 a_j 的方差。逼近向量 $\hat{\mathbf{x}}$ 的 l 个元素的总方差为

401

$$\sum_{j=1}^{l}\sigma_j^2 = \sum_{j=1}^{l}\lambda_j \tag{8.34}$$

在逼近误差向量 $\mathbf{x}-\hat{\mathbf{x}}$ 中的$(l-m)$个元素的总方差为

402

$$\sum_{j=l+1}^{m}\sigma_j^2 = \sum_{j=l+1}^{m}\lambda_j \tag{8.35}$$

特征值$\lambda_{l+1},\cdots,\lambda_m$ 是相关矩阵 $\mathbf{R}$ 的特征值中最小的$(m-l)$个特征值；在用于重构逼近向量 $\hat{\mathbf{x}}$ 的式(8.28)中丢弃了它们所对应的项。这些特征值越接近 0，降维(对 $\mathbf{x}$ 进行主分量分析所导致的结果)后保存原始数据中的信息量就越有效。因此，为了对输入数据进行维数缩减，我们计算输入数据向量的相关矩阵 $\mathbf{R}$ 的特征值和特征向量，然后将原始向量投影到 m 个优势特征值对应的特征向量生成的子空间。这种数据表示方法通常称为子空间分解(Oja, 1983)。

例 8.1 双变量数据集 为了说明主分量分析的应用，考虑双变量(二维)数据集的例子，如图 8-4，其中假设两个特征轴的标度近似相同。图中水平轴和垂直轴表示数据集的自然坐标轴。标号为 1 和 2 旋转坐标轴是应用这个数据集的主变量分析产生的结果。从图 8-4 可以看出数据集投影到 1 号轴上抓住了数据的主要特征，即具有双峰(即在它的结构上有两个聚类)的特点。的确，数据投影到轴 1 的方差比投影到别的轴上的大。相反，当映射到轴 2 时，数据内在的双峰特征完全模糊。

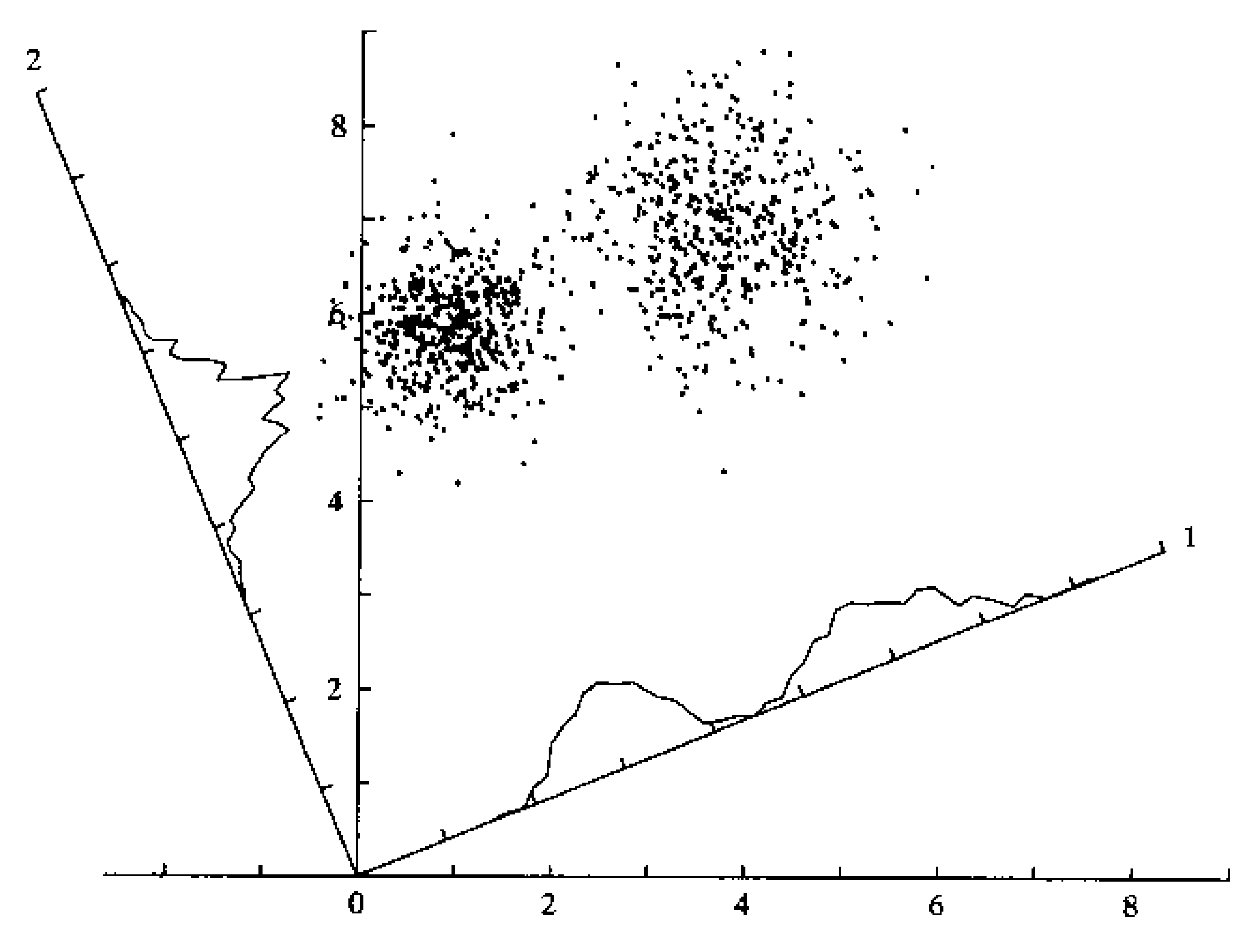


图 8-4 二维平面的一组数据，它们投影到两个轴 1 和 2 的密度图
投影到轴 1 有最大方差，清楚表明数据的双峰或聚类特征

从这个简单的例子中可以得到一个重要的结论。虽然，带有聚类结构的数据集在带有水平轴和垂直轴的二维平面图上很明显，但在实际中并不总是这样。在更一般的高维数据集中，可以想像数据固有的聚类结构被隐藏，要想看到它必须进行与主分量分析相似的统计分析(Linsker, 1988a)。 ■

403

8.4 基于 Hebb 的最大特征滤波器

自组织神经网络的行为和主分量分析的统计方法之间存在密切的联系。在本节，我们将通过建立一个著名的结果来证实这个关系：突触权值采用Hebb自适应规则的单个线性神经元能够形成关于输入分布第一个主分量的过滤器(Oja,1982)。

为了继续这个证明，先考虑如图8-5a所示的简单模型。该模型在模型输出为它的输入的线性组合这个意义下是线性的。神经元通过 m 个分别具有权值 $w_1, w_2, \ldots, w_m$ 的突触来接收 m 个输入信号 $x_1, x_2, \ldots, x_m$，模型的输出结果 y 为

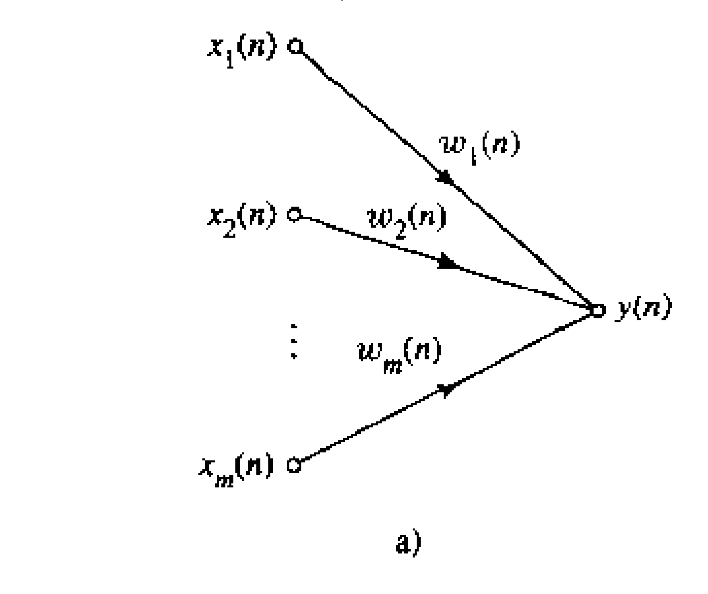
$$y = \sum_{i=1}^{m} w_i x_i \tag{8.36}$$

注意这里描述的情形，我们仅处理单个神经元，所以不需要用双下标表示网络突触权值。

根据 Hebb 学习的假设，当前突触信号 x_i 和后突触信号 y 一致时，突触权值随时间逐步加强。具体地，可写成

$$w_i(n+1) = w_i(n) + \eta y(n) x_i(n), \quad i = 1,2,\cdots,m \tag{8.37}$$

其中 n 表示离散时间，η 是学习率参数。



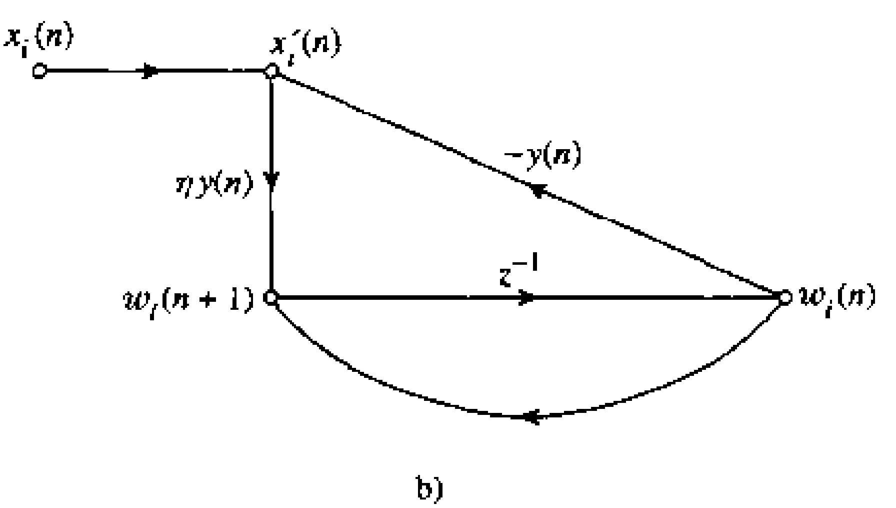


图 8-5 最大特征滤波器信号流图表示
a)式(8.36)的图 b)式(8.41)和(8.42)的图

404

但是，这个学习规则的基本形式会导致突触权值 w_i 无限地增大，这在现实上是不能接受的。在突触权值自适应学习规则中采用某种程度的饱和度或归一化，可以解决这个问题。利用归一化方法具有在神经元的突触权值间由于有限资源导致竞争的效果，从自组织的原则 2，这是稳定性的关键。从数学上来考虑，方便的归一化形式描述如下(Oja,1982)：

$$w_i(n+1) = \frac{w_i(n) + \eta y(n) x_i(n)}{\left(\sum_{i=1}^{m}[w_i(n) + \eta y(n) x_i(n)]^2\right)^{1/2}} \tag{8.38}$$

其中分母的求和是针对神经元的所有突触权值。假设学习率参数 η 很小，可以将式(8.38)展开成 η 的幂级数形式，所以写成

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n) w_i(n)] + O(\eta^2) \tag{8.39}$$

其中 $O(\eta^2)$项表示 η^2 或更高次部分。因为 η 很小，可以忽略这一项，因此近似式(8.38)到 η 的一阶项如下：

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n) w_i(n)] \tag{8.40}$$

式(8.40)右端的项 $y(n)x_i(n)$表示突触权值通常的 Hebb 修改变，这符合自组织原则 1 描绘的自放大效果。依据原则 2，该式中含有负项 $-y(n)w_i(n)$导致稳定；它修改输入 $x_i(n)$成一种依赖于相应突触权值 $w_i(n)$和输出 $y(n)$的形式，表示为

$$x'_i(n) = x_i(n) - y(n)w_i(n) \tag{8.41}$$

$x'_i(n)$可以视为第 i 个突触的*有效输入*。我们可以由式(8.41)的定义重写式(8.40)的学习规则如下：

$$w_i(n+1) = w_i(n) + \eta y(n)x'_i(n) \tag{8.42}$$

405 神经元的整体操作可由两个信号流图的组合来表示，如图 8-5 所示。根据式(8.36)，图 8-5a 的信号流图表明输出 $y(n)$依赖于权值$w_1(n), w_2(n), \cdots, w_m(n)$。图 8-5b 的信号流图提供式(8.41)和(8.42)的图像；图中的传递参数 z^{-1}表示单位延迟操作符。在图 8-5a 中所产生的输出 $y(n)$在图 8-5b 中作为传递系数。图 8-5b 清楚地展示作用于神经元的内部反馈的下列两种形式

- 根据外部输入 $x_i(n)$，自放大的正反馈使得突触权值 $w_i(n)$增加。
- 由于 $-y(n)$的负反馈控制 $w_i(n)$的增大，因此导致突触权值 $w_i(n)$的稳定。

乘积项 $-y(n)w_i(n)$与在学习规则中经常用到的*遗忘因子*或*泄漏因子*有关，但存在差别：对于较强的响应 $y(n)$，遗忘因子变得更加显著。这种控制现象有神经生物上的支持(Stent, 1973)。

算法的矩阵形式

为了描述上的方便，令

$$\mathbf{x}(n) = [x_1(n), x_2(n), \cdots, x_m(n)]^T \tag{8.43}$$

和

$$\mathbf{w}(n) = [w_1(n), w_2(n), \cdots, w_m(n)]^T \tag{8.44}$$

输入向量 $\mathbf{x}(n)$和突触权值向量 $\mathbf{w}(n)$通常都是随机向量的实现。用这个向量符号可以重写式(8.36)为内积形式如下：

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{w}^T(n)\mathbf{x}(n) \tag{8.45}$$

同样地，可以重写式(8.40)为

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n)[\mathbf{x}(n) - y(n)\mathbf{w}(n)] \tag{8.46}$$

将式(8.45)代入(8.46)得

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n)\mathbf{w}(n)] \tag{8.47}$$

式(8.47)所示的学习算法为*非线性随机差分方程*，这使得该算法的收敛性分析在数学上很难进行。为了得到收敛性分析，我们先简单介绍随机逼近算法收敛分析的一般工具。

渐进稳定性定理

式(8.47)表示的自组织算法是一般的随机逼近算法

406
$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)h(\mathbf{w}(n), \mathbf{x}(n)), \qquad n = 0,1,2,\cdots, \tag{8.48}$$

的一种特殊形式。序列 $\eta(\cdot)$是一个正的标量序列。

更新函数(update function) $h(\cdot,\cdot)$是具有某些正则性条件的确定性函数。$h(\cdot,\cdot)$和标量序列 $\eta(\cdot)$完全确定算法的具体结构。

这里描述的过程的目的是将随机非线性差分方程(8.48)和*确定性的常微分方程*(ODE)联系起来。于是微分方程的稳定性和算法的收敛性联系在一起。这个过程是很一般的工具，具有很广的用途。这分别由 Ljung(1977)与 Kushner and Clark(1978)独立提出，但用不同方法[2]。

为了开始，过程假设式(8.48)描述的随机逼近算法满足下面的条件(用我们的术语表示)：

1. $\eta(n)$为下降的正实数序列，使得我们有

(a) $$\sum_{n=1}^{\infty}\eta(n) = \infty \tag{8.49}$$

(b) $$\sum_{n=1}^{\infty}\eta^{p}(n) < \infty \qquad 对\ p > 1 \tag{8.50}$$

(c) $$\eta(n)\to 0 \qquad 当\ n\to\infty \tag{8.51}$$

2. 参数向量序列(突触权值)$\mathbf{w}(\cdot)$有界的概率为1。

3. 更新函数 $h(\mathbf{w},\mathbf{x})$对 $\mathbf{w}$ 和 $\mathbf{x}$ 连续可微，且其导数在时间上一致有界。

4. 对每个 $\mathbf{w}$ 存在极限

$$\bar{h}(\mathbf{w}) = \lim_{n\to\infty} E[h(\mathbf{w},\mathbf{X})] \tag{8.52}$$

统计期望运算符 E 对随机向量 $\mathbf{X}$ 操作，$\mathbf{X}$ 的实现由 $\mathbf{x}$ 表示。

5. 常微分方程

$$\frac{d}{dt}\mathbf{w}(t) = \bar{h}(\mathbf{w}(t)) \tag{8.53}$$

具有局部渐进稳定解(Lyapunov 意义下)，其中 t 表示连续时间，Lyapunov 意义的稳定性在第14章讨论。

6. 令 $\mathbf{q}_1$ 表示式(8.53)的解，具有吸引域$\mathscr{B}(\mathbf{q})$；吸引域在第14章定义。那么参数向量 $\mathbf{w}(n)$以概率1经常无穷次进入吸引域$\mathscr{B}(\mathbf{q})$的紧子集$\mathscr{A}$。

407

这里描述的6个条件都是合理的。具体地，条件1(a)是使算法在任意初始条件下能够将估计值移到期望极限的必要条件。条件1(b)给定 $\eta(n)$趋向0有多快的条件；这比常用的条件

$$\sum_{n=1}^{\infty}\eta^{2}(n) < \infty$$

的限制更少。条件4使一个微分方程与式(8.48)所示的算法相联系成为可能的基本假设。

考虑递归等式(8.48)描述的随机逼近算法，它满足假设1至6。那么我们可以陈述这类随机逼近算法的渐进稳定性定理如下(Ljung,1977;Kushner and Clark,1978)：

$$\lim_{n\to\infty}\mathbf{w}(n) = \mathbf{q}_1 \qquad 以概率1经常无限地成立 \tag{8.54}$$

但是，我们强调这里描述过程虽然提供关于算法(8.48)的渐进性质的信息，但它并没有告诉我们迭代次数 n 应该选多大才能使分析结果可用。此外，在利用式(8.48)算法解决时变参数向量的问题时，要求

$$\eta(n)\to 0 \qquad 当\ n\to\infty$$

是不可行的，这由条件1(c)规定。我们可以通过指定 η 的一个很小的正数来克服后面这个困难，指定的数的大小由应用决定。随机逼近算法在神经网络的实际应用中经常这样做。

最大特征滤波器的稳定性分析

在稳定性的ODE方法中，我们具备研究由式(8.46)表示的递归算法的收敛行为所需的工具，正如这里的描述，这个递归算法与最大特征滤波器相关。

为了满足渐进稳定性定理的条件1，我们令

$$\eta(n) = \frac{1}{n}$$

其次，从式(8.47)注意更新函数 $h(\mathbf{w}, \mathbf{x})$ 由

$$\begin{aligned} h(\mathbf{w},\mathbf{x}) &= \mathbf{x}(n)y(n) - y^2(n)\mathbf{w}(n) \\ &= \mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - [\mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n)]\mathbf{w}(n) \end{aligned} \tag{8.55}$$

定义，很显然它满足定理条件3。在更新函数 $h(\mathbf{w},\mathbf{X})$ 中利用随机向量 $\mathbf{X}$ 的一个实现 $\mathbf{x}$ 得到式(8.55)。由条件4，我们对 $\mathbf{X}$ 求取 $h(\mathbf{w},\mathbf{X})$ 的期望值，从而可写成

$$\begin{aligned} h &= \lim_{n\to\infty} E[\mathbf{X}(n)\mathbf{X}^T(n)\mathbf{w}(n) - (\mathbf{w}^T(n)\mathbf{X}(n)\mathbf{X}^T(n)\mathbf{w}(n))\mathbf{w}(n)] \\ &= \mathbf{R}\mathbf{w}(\infty) - [\mathbf{w}^T(\infty)\mathbf{R}\mathbf{w}(\infty)]\mathbf{w}(\infty) \end{aligned} \tag{8.56}$$

其中 $\mathbf{R}$ 是随机向量 $\mathbf{X}$ 表示的随机过程的相关矩阵，$\mathbf{w}(\infty)$ 是突触权值向量的极限值。

由条件5并根据式(8.53)和(8.56)，我们寻找非线性微分方程

408

$$\frac{d}{dt}\mathbf{w}(t) = \bar{h}(\mathbf{w}(t)) = \mathbf{R}\mathbf{w}(t) - [\mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t)]\mathbf{w}(t) \tag{8.57}$$

的稳定点。根据相关矩阵 $\mathbf{R}$ 特征向量的完全正交集将 $\mathbf{w}(t)$ 展开成

$$\mathbf{w}(t) = \sum_{k=1}^{m}\theta_k(t)\mathbf{q}_k \tag{8.58}$$

其中 $\mathbf{q}_k$ 是 $\mathbf{R}$ 的第 k 个归一化特征向量，系数 $\theta_k(t)$ 是向量 $\mathbf{w}(t)$ 在 $\mathbf{q}_k$ 上的时变投影。将式(8.58)代入式(8.57)，并使用基本定义

$$\mathbf{R}\mathbf{q}_k = \lambda_k\mathbf{q}_k$$

和

$$\mathbf{q}_k^T\mathbf{R}\mathbf{q}_k = \lambda_k$$

其中 λ_k 是与 $\mathbf{q}_k$ 相关的特征值，最后我们得到

$$\sum_{k=1}^{m}\frac{d\theta_k(t)}{dt}\mathbf{q}_k = \sum_{k=1}^{m}\lambda_k\theta_k(t)\mathbf{q}_k - \left[\sum_{l=1}^{m}\lambda_l\theta_l^2(t)\right]\sum_{k=1}^{m}\theta_k(t)\mathbf{q}_k \tag{8.59}$$

等价地，我们可写成

$$\frac{d\theta_k(t)}{dt} = \lambda_k\theta_k(t) - \theta_k(t)\sum_{l=1}^{m}\lambda_l\theta_l^2(t), k = 1,2,\cdots,m \tag{8.60}$$

从而我们将式(8.48)的随机逼近算法的收敛性分析归结为涉及主模式(principal mode)$\theta_k(t)$的常微分方程组(8.60)的系统稳定性分析。

依赖于对下标 k 所赋给的值，可分为两种情况。情况Ⅰ对应于 $1<k\leqslant m$。情况Ⅱ对应于 $k=1$；m 为 $\mathbf{x}(n)$ 和 $\mathbf{w}(n)$ 的维数。依次考虑这两种情况。

情况Ⅰ $1<k\leqslant m$。为处理这种情况我们定义

$$\alpha_k(t) = \frac{\theta_k(t)}{\theta_1(t)}, \qquad 1 < k \leqslant m \tag{8.61}$$

首先假设 $\theta_1(t)\neq 0$，若初始值 $\mathbf{w}(0)$ 随机选取，这以概率1为真。对式(8.61)两边对时间 t 求导数得到

$$\begin{aligned} \frac{d\alpha_k(t)}{dt} &= \frac{1}{\theta_1(t)}\frac{d\theta_k(t)}{dt} - \frac{\theta_k(t)}{\theta_1^2(t)}\frac{d\theta_1(t)}{dt} \\ &= \frac{1}{\theta_1(t)}\frac{d\theta_k(t)}{dt} - \frac{\alpha_k(t)}{\theta_1(t)}\frac{d\theta_1(t)}{dt}, \qquad 1 < k \leqslant m \end{aligned} \tag{8.62}$$

其次，将式(8.60)代入式(8.62)，利用式(8.61)的定义并化简结果，我们得到

$$\frac{d\alpha_k(t)}{dt} = -(\lambda_1 - \lambda_k)\alpha_k(t), \qquad 1 < k \leqslant m \tag{8.63}$$

409

假设相关矩阵 **R** 的特征值互不相同且按降序排列，则有

$$\lambda_1 > \lambda_2 > \cdots > \lambda_k > \cdots > \lambda_m > 0 \tag{8.64}$$

由此推知特征值之差 $\lambda_1 - \lambda_k$ 为正，在式(8.63)中表示一个时间常数的倒数。所以，从情况 I 发现：

$$\alpha_k(t) \to 0 \qquad 当\ t \to \infty \qquad 对于\ 1 < k \leqslant m \tag{8.65}$$

情况 II $k = 1$。从式(8.60)可知，这第二种情况由微分方程

$$\begin{aligned}\frac{d\theta_1(t)}{dt} &= \lambda_1\theta_1(t) - \theta_1(t)\sum_{l=1}^{m}\lambda_l\theta_l^2(t) = \lambda_1\theta_1(t) - \lambda_1\theta_1^3(t) - \theta_1(t)\sum_{l=2}^{m}\lambda_l\theta_l^2(t) \\ &= \lambda_1\theta_1(t) - \lambda_1\theta_1^3(t) - \theta_1^3(t)\sum_{l=2}^{m}\lambda_l\alpha_l^2(t)\end{aligned} \tag{8.66}$$

描述。然而，从情况 I 我们知道，当 $t \to \infty$ 时，对于 $l \neq 1$，$\alpha_l \to 0$。因此，当 t 趋向无穷大时，式(8.66)右端的最后一项接近 0。忽略此项，式(8.66)简化为

$$\frac{d\theta_1(t)}{dt} = \lambda_1\theta_1(t)[1 - \theta_1^2(t)] \qquad 对\ t \to \infty \tag{8.67}$$

但是必须强调，只在渐进意义下式(8.67)成立。

方程(8.67)表示自治系统(即系统不显式依赖于时间)。这样一种系统的稳定性最好由称为 Lyapunov 函数的正定函数处理，Lyapunov 函数的具体地处理细节在第 14 章介绍。令 **s** 表示自治系统的状态向量，$V(t)$ 表示系统的 Lyapunov 函数。如果满足下列条件，则系统的平衡状态 $\bar{\mathbf{s}}$ 是渐进稳定的：

$$\frac{d}{dt}V(t) < 0 \qquad 对\ \mathbf{s} \in \mathcal{U} - \mathbf{s}$$

其中 $\mathcal{U}$ 为 $\bar{\mathbf{s}}$ 的小邻域。

对当前的问题，我们断言微分方程(8.67)有一个由

$$V(t) = [\theta_1^2(t) - 1]^2 \tag{8.68}$$

定义的 Lyapunov 函数。为了证实这个断言，必须证明 $V(t)$ 需要满足下面两个条件：

1. $\dfrac{dV(t)}{dt} < 0$ 对所有 t (8.69)

2. $V(t)$ 具有最小值 (8.70)

410

在式(8.68)中对 t 求导得

$$\frac{dV(t)}{dt} = 4\theta_1(t)[\theta_1(t) - 1]\frac{d\theta_1}{dt} = -4\lambda_1\theta_1^2(t)[\theta_1^2(t) - 1]^2 \qquad 对于\ t \to \infty \tag{8.71}$$

其中在第二个等式利用了式(8.67)。因为特征值 λ_1 是正的，从式(8.71)发现，当 t 趋近无穷大时，式(8.69)的条件为真。此外，从式(8.71)知 $V(t)$ 在 $\theta_1(t) = \pm 1$ 处具有最小值(即 $\dfrac{dV(t)}{dt} = 0$)，所以式(8.70)的条件也满足。因此我们可以用下列陈述结束情况 II 的分析：

$$\theta_1(t) \to \pm 1 \qquad 当\ t \to \infty \tag{8.72}$$

根据式(8.72)中描述的结果和式(8.71)的定义，可以重新陈述式(8.65)中情况 I 的结果的最终形式：

$$\theta_k(t) \to 0 \qquad \text{当 } t \to \infty \qquad \text{对于 } 1 < k \leq m \tag{8.73}$$

从情况Ⅰ和Ⅱ的分析作出的全面结论是两方面的：

- 式(8.47)描述的随机逼近算法仅主模式收敛于 $\theta_1(t)$，算法的其他所有模式将衰减为0。
- 模式 $\theta_1(t)$收敛于 ± 1。

因此，渐进稳定性定理的条件5满足。特别，依据式(8.58)的展开式，可以正式地陈述

$$\mathbf{w}(t) \to \mathbf{q}_1 \qquad \text{当 } t \to \infty$$

其中 $\mathbf{q}_1$ 是相关矩阵 $\mathbf{R}$ 的最大特征值 λ_1 对应的归一化特征向量。

根据渐进稳定性定理的条件6，我们必须证明对存在所有向量集合的子集$\mathcal{A}$满足

$$\lim_{n \to \infty} \mathbf{w}(n) = \mathbf{q}_1 \qquad \text{以概率1无限地经常成立}$$

为了这样做，我们必须满足条件2，这可通过硬性限制 $\mathbf{w}(n)$的项，使它们的幅度值小于阈值 a。我们可以定义 $\mathbf{w}(n)$的范数为

411

$$\| \mathbf{w}(n) \| = \max_j |w_j(n)| \leq a \tag{8.74}$$

令$\mathcal{A}$是 $\mathbb{R}^m$ 的压缩子集，由一个范数小于等于 a 的向量集定义。可以直接证明(Sanger, 1989b)。

如果 $\| \mathbf{w}(n) \| \leq a$，且常数 a 足够大，则 $\| \mathbf{w}(n+1) \| < \| \mathbf{w}(n) \|$ 以概率1成立。

于是，随着迭代次数 n 的增大，$\mathbf{w}(n)$将最终进入$\mathcal{A}$内并以概率1留在$\mathcal{A}$内。因为吸引域$\mathcal{B}(\mathbf{q}_1)$包括所有有界范数的向量，因此有$\mathcal{A} \in \mathcal{B}(\mathbf{q}_1)$。换句话说，条件6满足。

现在渐进稳定性定理的所有6个条件都满足了，因此证明(满足前面提到的假设)随机逼近算法(8.47)将使 $\mathbf{w}(n)$以概率1收敛于特征向量 $\mathbf{q}_1$，$\mathbf{q}_1$ 是与相关矩阵 $\mathbf{R}$ 的最大特征值 λ_1 对应的特征向量。这不仅是算法的固定点，而且是惟一的渐进稳定点。

基于 Hebb 最大特征滤波器的性质小结

刚才给出的收敛分析只证明由式(8.39)或等价地式(8.46)的自组织学习规则控制的单个线性神经元自适应地抽取平稳输入的第一个主分量。这第一个主分量对应于随机向量 $\mathbf{X}(n)$ 的相关矩阵的最大特征值 λ_1；事实上 λ_1 与模型输出 $y(n)$的方差有关，如下所示。

令 $\sigma^2(n)$表示随机变量 $Y(n)$的方差，$y(n)$表示 $Y(n)$的一次实现，即

$$\sigma^2(n) = E[Y^2(n)] \tag{8.75}$$

其中由于输入均值为零，$Y(n)$具有0均值。在式(8.46)中令 $n \to \infty$ 并且利用 $\mathbf{w}(n)$趋向于 $\mathbf{q}_1$ 的事实，我们得到

$$\mathbf{x}(n) = y(n)\mathbf{q}_1 \qquad \text{对 } n \to \infty$$

利用这个关系，可以证明当迭代次数 n 趋向于∞时，方差 $\sigma^2(n)$趋向于 λ_1；参见习题8.2。

总之，其运行由式(8.46)描述的基于 Hebb 的线性神经元以概率1收敛于一个固定点，它具有如下的特征(Oja, 1982)：

1. 模型输出的方差趋向于相关矩阵 $\mathbf{R}$ 的最大特征值，表示为

$$\lim_{n \to \infty} \sigma^2(n) = \lambda_1 \tag{8.76}$$

2. 模型的突触权值向量趋向相关的特征向量，表示为

$$\lim_{n\to\infty} \mathbf{w}(n) = \mathbf{q}_1 \tag{8.77}$$

且

$$\lim_{n\to\infty} \| \mathbf{w}(n) \| = 1 \tag{8.78}$$

这些结果均假设相关矩阵 **R** 是正定的，且 **R** 的最大特征值 λ_1 的重数为 1。这些结果也适用于具有 $\lambda_1 > 0$ 且重数为 1 的非负定相关矩阵 **R**。 [412]

例 8.2　匹配滤波器　考虑随机向量 $\mathbf{X}(n)$ 组成如下：

$$\mathbf{X}(n) = \mathbf{s} + \mathbf{V}(n)$$

其中 $\mathbf{s}$ 为固定单位向量，表示信号分量，$\mathbf{V}(n)$ 为零均值的白噪声分量。输入向量的相关矩阵为

$$\mathbf{R} = E[\mathbf{X}(n)\mathbf{X}^T(n)] = \mathbf{s}\mathbf{s}^T + \sigma^2\mathbf{I}$$

其中 σ^2 是噪声向量 $\mathbf{V}(n)$ 元素的方差，$\mathbf{I}$ 是单位矩阵。因此相关矩阵 **R** 的最大特征值

$$\lambda_1 = 1 + \sigma^2$$

对应的特征向量 $\mathbf{q}_1$ 为

$$\mathbf{q}_1 = \mathbf{s}$$

容易证明，在这种情况下这个解满足特征值问题

$$\mathbf{R}\mathbf{q}_1 = \lambda_1 \mathbf{q}_1$$

因此，对于本例描述的情况，自组织线性神经元(从收敛到它的稳定条件)充当一个匹配的滤波器，其冲击响应(由突触权值表示)与输入向量 $\mathbf{X}(n)$ 的信号分量 $\mathbf{s}$ 匹配。 ■

8.5　基于 Hebb 的主分量分析

上一节中基于 Hebb 的最大特征滤波器抽出输入的第一个主分量。这个单线性神经元模型可以扩展到单层线性神经元的前馈网络，目的在于对输入进行任意大小的主分量分析(Sanger, 1989b)。

具体地，考虑如图 8-6 所示的前馈网络。假设具有下面两个结构属性：

1. 网络输出层的每个神经元是线性的。

2. 网络有 m 个输入和 l 个输出，它们都是指定的。另外，网络输出少于输入(即 $l < m$)。

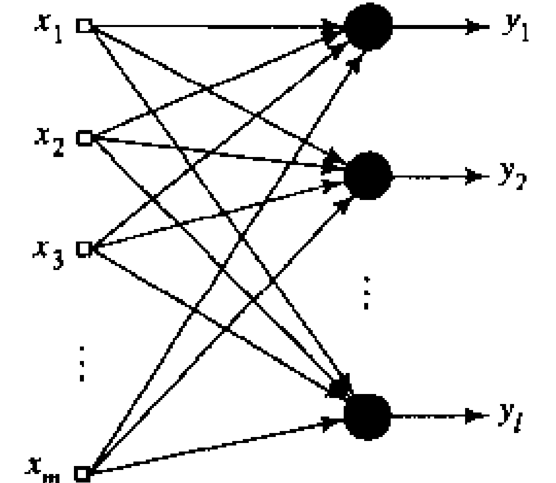


图 8-6　仅有单层计算节点的前馈网络

[413]

网络接受训练的仅有突触权值集 $\{w_{ji}\}$，它们将输入层的源节点 i 和输出层计算节点 j 连接起来，其中 $i = 1, 2, \cdots, m$ 和 $j = 1, 2, \cdots, l$。

在时刻 n 神经神经元 j 对输入集 $\{x_i(n) | i = 1, 2, \cdots, m\}$ 的响应所产生的输出 $y_j(n)$ 由下式给出(参看图 8-7a)：

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n), \qquad j = 1, 2, \cdots, l \tag{8.79}$$

根据 Hebb 学习的广义形式，修改突触权值 $w_{ji}(n)$ 采用下式(Sanger, 1989b)：

$$\Delta w_{ji}(n) = \eta\left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right], \qquad \begin{array}{l} i = 1, 2, \cdots, m \\ j = 1, 2, \cdots, l \end{array} \tag{8.80}$$

其中 $\Delta w_{ji}(n)$ 是在时刻 n 对 $w_{ji}(n)$ 的修改，η 是学习率。对于一层含有 l 个神经元的式

(8.80)所示的广义 Hebb 算法(generalized Hebbian algorithm, GHA)包括上一节式(8.39)的算法为其特殊情况，即 $j=1$。

为对该算法的行为进行分析，将式(8.80)重新写成

$$\Delta w_{ji}(n)=\eta y_j(n)[x'_i(n)-w_{ji}(n)y_j(n)],\quad \begin{array}{l} i=1,2,\cdots,m \\ j=1,2,\cdots,l \end{array} \tag{8.81}$$

的形式，其中 $x'_i(n)$为输入向量 $\mathbf{x}(n)$的第 i 个分量的修改形式；它是下标 j 的函数，表示为

$$x'_i(n)=x_i(n)-\sum_{k=1}^{j-1}w_{ki}(n)y_k(n) \tag{8.82}$$

对某个指定的神经元 j，式(8.81)表示的算法与上一节式(8.39)表示的算法在数学形式上完全相同，只是将 $x_i(n)$变成了它的修改值 $x'_i(n)$。可以进一步将公式(8.80)重新写成 Hebb 的学习假设对应的形式，表示成

$$\Delta w_{ji}(n)=\eta y_j(n)x''_i(n) \tag{8.83}$$

其中

$$x''_i(n)=x'_i-w_{ji}(n)y_j(n) \tag{8.84}$$

注意

$$w_{ji}(n+1)=w_{ji}(n)+\Delta w_{ji}(n) \tag{8.85}$$

[414] 和

$$w_{ji}(n)=z^{-1}[w_{ji}(n+1)] \tag{8.86}$$

其中 z^{-1}是单位延迟操作符，我们可以构建广义 Hebb 算法的信号流图，如图 8-7b 所示。从图中看出只要其公式由式(8.85)描述，则算法适合于实现的局部形式。同时注意在图 8-7b 的信号流图中表示反馈的 $y_j(n)$由式(8.79)决定；它的信号流图表示在图 8-7a 给出。

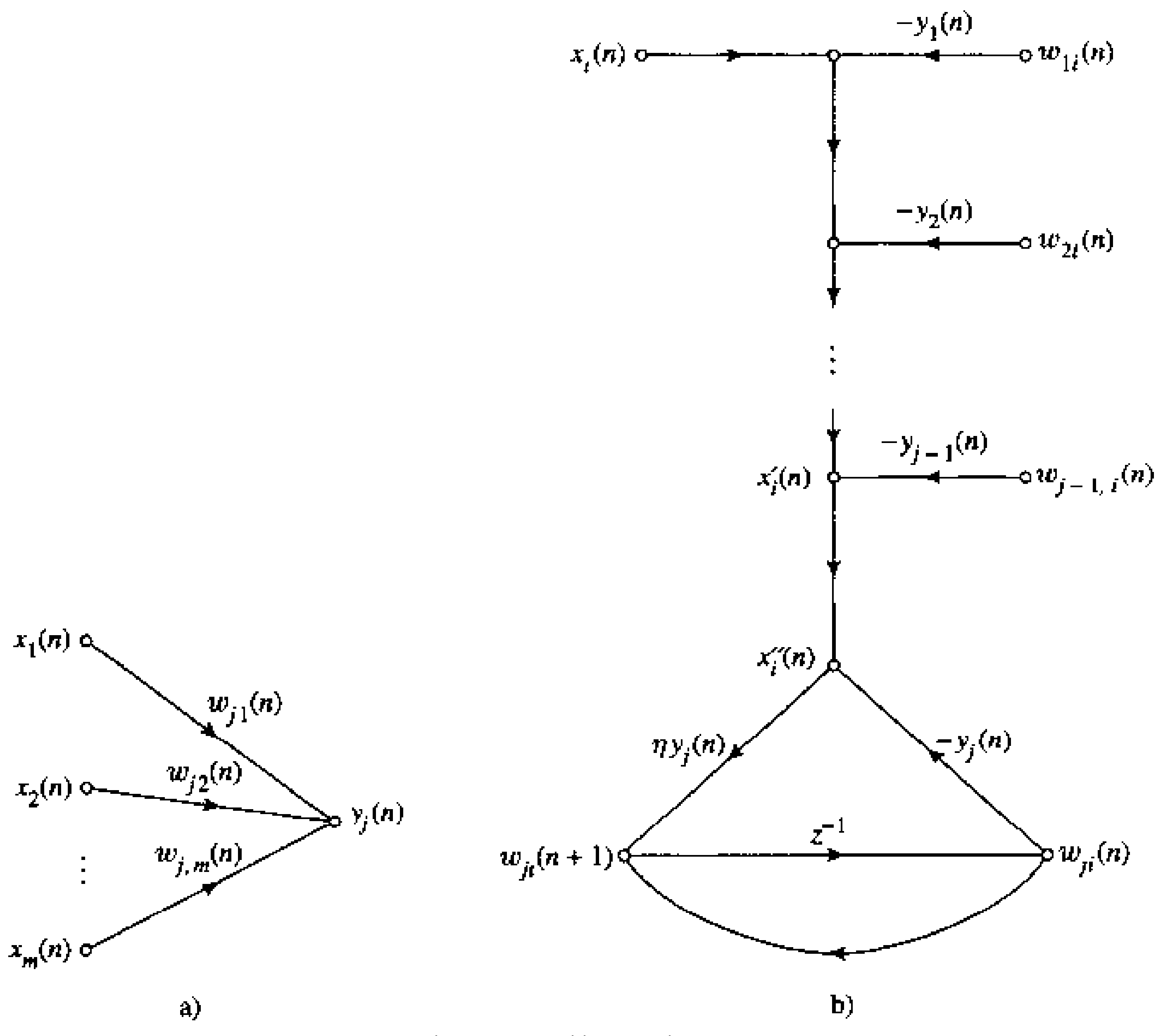


图 8-7 广义 Hebb 算法的信号流图表示

a)式(8.79)的图 b)式(8.80)至(8.81)的图

为了有助于理解广义 Hebb 算法实际上如何操作，我们首先利用矩阵形式重写式(8.81)定义的算法如下：

$$\Delta \mathbf{w}_j(n) = \eta y_j(n)\mathbf{x}'(n) - \eta y_j^2(n)\mathbf{w}_j(n), \qquad j = 1,2,\cdots,l \tag{8.87}$$

其中

$$\mathbf{x}'(n) = \mathbf{x}(n) - \sum_{k=1}^{j-1}\mathbf{w}_k(n)y_k(n) \tag{8.88}$$

415

向量 $\mathbf{x}'(n)$为输入向量的修正形式。基于式(8.87)给出的表示，我们得到下面的观察结果(Sanger,1989b)：

1. 对于图 8-6 的前馈网络中的第一个神经元，我们有

$$j = 1: \qquad \mathbf{x}'(n) = \mathbf{x}(n)$$

这种情况下，广义 Hebb 算法相当于上一节的一个神经元的式(8.46)。由 8.4 节的描述，我们已经知道这个神经元将发现输入向量的第一个主分量。

2. 对于图 8-6 中的第 2 个神经元，我们写出

$$j = 2: \qquad \mathbf{x}'(n) = \mathbf{x}(n) - \mathbf{w}_1(n)y_1(n)$$

如果第一个神经元已经收敛于第一个主分量，则第二个神经元看到一个输入向量 $\mathbf{x}'(n)$，从其中已经去掉相关矩阵 **R** 的第一个特征向量。因此第二个神经元抽取的是 $\mathbf{x}'(n)$的第一个主分量，相当于原来输入向量 $\mathbf{x}(n)$的第二个主分量。

3. 对于第 3 个神经元，我们写出

$$j = 3: \qquad \mathbf{x}'(n) = \mathbf{x}(n) - \mathbf{w}_1(n)y_1(n) - \mathbf{w}_2(n)y_2(n)$$

假设前两个神经元已经分别收敛于第一个和第二个主分量，如前面两步的解释一样。第三个神经元的输入向量为 $\mathbf{x}'(n)$，从其中已经去掉相关矩阵 **R** 的前两个特征向量。因此第三个神经元抽取的是 $\mathbf{x}'(n)$的第一个主分量，相当于原来输入向量 $\mathbf{x}(n)$的第三个主分量。

4. 对于图 8-6 的前馈网络中剩下的神经元，继续执行上述过程。显然根据式(8.81)的广义 Hebb 算法训练的网络的每个输出代表对应于输入向量相关矩阵的某一特征向量的响应，并且这些输出按特征值递减排序。

这个计算特征向量的方法与通称为 Hotelling 的紧缩技术(Hotelling's deflation technique)相似(Kreyszig,1988)；它类似于 Gram-Schmidt 正交化过程(Strang,1980)。

这里所给的一个神经元接一个神经元地描述仅仅是为了简化解释。实际上，在广义 Hebb 算法中所有的神经元趋于同时收敛。

收敛性考虑

令 $\mathbf{W}(n) = \{w_{ji}(n)\}$表示图 8-6 所示前馈网络的一个 $l \times m$ 的权值矩阵，即

$$\mathbf{W}(n) = [\mathbf{w}_1(n), \mathbf{w}_2(n), \cdots, \mathbf{w}_l(n)]^T \tag{8.89}$$

令广义 Hebb 算法的学习率参数 η 取时变形式 $\eta(n)$，限制条件为

$$\lim_{n\to\infty}\eta(n) = 0 \qquad 和 \qquad \sum_{n=0}^{\infty}\eta(n) = \infty \tag{8.90}$$

可以将算法重新写成矩阵形式

$$\Delta \mathbf{W}(n) = \eta(n)\{\mathbf{y}(n)\mathbf{x}^T(n) - \mathrm{LT}[\mathbf{y}(n)\mathbf{y}^T(n)]\mathbf{W}(n)\} \tag{8.91}$$

416

其中 LT[·]为下三角算子，它把矩阵对角线上方的所有元素置为 0，从而使矩阵成为下三角矩阵。在这些条件下以及采用 8.4 节所作的假设，则 GHA 算法收敛性证明的过程与上节关

于最大特征滤波器的收敛证明相似。因此我们可以陈述下面的定理(Sanger,1989b)：

如果权值矩阵 $\mathbf{W}(n)$在时间步 $n=0$ 时随机赋值，则式(8.91)所描述的广义 Hebb 算法以概率1收敛于固定点，且 $\mathbf{W}^T(n)$趋于一个矩阵，该矩阵的列分别为 $m \times 1$ 输入向量的 $m \times m$ 的相关矩阵 $\mathbf{R}$ 的前 l 个特征向量，按特征值的降序排列。

这个定理的实际价值在于，当对应特征值互不相同时它保证广义 Hebb 算法能够找到相关矩阵 $\mathbf{R}$ 的前 l 个特征向量。同样重要的是，我们不需要计算相关矩阵 $\mathbf{R}$，$\mathbf{R}$ 的前 l 个特征向量可直接由输入向量计算。特别是如果输入空间的维数 m 很大，而要求与 $\mathbf{R}$ 最大的 l 个最大特征值对应的特征向量的数目只是 m 的一小部分，则导致的计算节省可能是巨大的。

收敛定理是用时变学习率参数 $\eta(n)$表示的。实际上，学习率参数只能选择一个很小的固定常数 η，这样才能保证在 η 阶的突触权值的均方误差意义下收敛。

在 Chatterjee et al.(1998)中，研究式(8.91)描述的 GHA 算法的收敛性质。那里给出的分析表明，η增加将导致收敛速度加快，同时渐进均方误差也会增大；这在直观上也是符合的。除此之外，该论文对计算的精确性和学习速度之间的折中作了清楚的描述。

广义 Hebb 算法的最优性

假设在极限时写成

$$\Delta\mathbf{w}_j(n) \to \mathbf{0} \text{ 和 } \mathbf{w}_j(n) \to \mathbf{q}_j \quad \text{当 } n \to \infty \qquad \text{对于 } j = 1,2,\cdots,l \tag{8.92}$$

并且有

$$\|\mathbf{w}_j(n)\| = 1 \qquad \text{对于所有 } j \tag{8.93}$$

那么在图 8-5 所示的前馈网络中，神经元的突触权值向量的极限值$\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_l$ 表示相关矩阵 $\mathbf{R}$ 的前 l 个特征值对应的归一化特征向量，按特征值的降序排列。在平衡时可写为

[417]

$$\mathbf{q}_j^T\mathbf{R}\mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \tag{8.94}$$

其中 $\lambda_1 > \lambda_2 > \cdots > \lambda_l$。

对于神经元 j 的输出，我们有极限值

$$\lim_{n\to\infty} y_j(n) = \mathbf{x}^T(n)\mathbf{q}_j = \mathbf{q}_j^T\mathbf{x}(n) \tag{8.95}$$

令 $Y_j(n)$用表示一个随机变量，其实现记为输出 $y_j(n)$。在平衡时随机变量 $Y_j(n)$和 $Y_k(n)$的互相关为：

$$\lim_{n\to\infty} E[Y_j(n)Y_k(n)] = E[\mathbf{q}_j^T\mathbf{X}(n)\mathbf{X}^T(n)\mathbf{q}_k] = \mathbf{q}_j^T\mathbf{R}\mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \tag{8.96}$$

因此，我们可以陈述：在平衡时式(8.91)的广义 Hebb 算法充当输入数据的特征分析器。

令 $\hat{\mathbf{x}}(n)$表示输入向量 $\mathbf{x}(n)$的特定值，对于这个值，式(8.92)的极限条件对 $j = l-1$ 是满足的。因此，从式(8.80)的矩阵形式，我们发现在极限形式

$$\hat{\mathbf{x}}(n) = \sum_{k=1}^{l} y_k(n)\mathbf{q}_k \tag{8.97}$$

这意味着给定两组值，即图 8-5 的前馈网络中神经元的突触权值向量的极限值$\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_l$ 和相应的输出 $y_1(n), y_2(n), \cdots, y_l(n)$，我们可以构造输入向量 $\mathbf{x}(n)$的线性最小平方

估计$\hat{\mathbf{x}}(n)$。实际上，如图8-8所描绘的式(8.97)的公式可视为一种数据重建。注意根据在8.3节中的讨论，这种数据重建的方法导致逼近误差向量和估计$\hat{\mathbf{x}}(n)$正交。

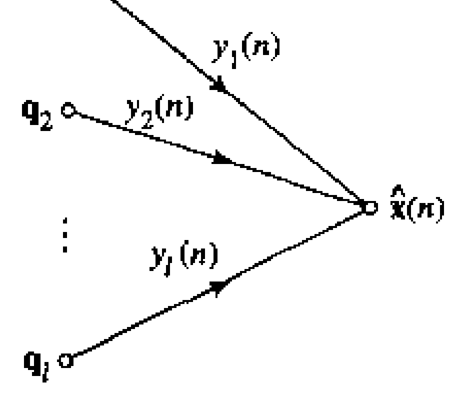


图8-8　如何计算重建向量$\hat{\mathbf{x}}$的信号流图表示

GHA 小结

广义 Hebb 算法(GHA)所涉及的计算很简单，可以小结如下：

[418]

1. 在时刻 $n=1$ 时，初始化网络突触权值 w_{ji}，使其取一个小的随机数。对学习率参数 η 赋给一个小的正数。



2. 对于$n=1, j=1,2,\cdots,l$ 和 $i=1,2,\cdots,m$ 计算

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n)x_i(n)$$

$$\Delta w_{ji}(n) = \eta\left[y_j(n)x_i(n) - y_j(n)\sum_{k=1}^{j} w_{ki}(n)y_k(n)\right]$$

其中，$x_i(n)$是 $m\times 1$ 输入向量 $\mathbf{x}(n)$的第 i 个分量，l 是期望的主分量个数。

3. n 增加 1($n=n+1$)，转到第2步，并继续执行直到 w_{ji}达到稳态值。对较大的 n，神经元 j 的突触权值w_{ji}收敛于输入向量 $\mathbf{x}(n)$的相关矩阵的第j 个特征值对应特征向量的第i 个分量。

8.6 计算机实验：图像编码

通过用广义 Hebb 学习算法解决图像编码问题完成对该算法的讨论。

图 8-9b 表示用于训练的一个双亲图像；该图像强调边缘信息。它被数字化为 256×256 的图像，分为 256 个灰度等级。利用一个具有 8 个神经元的单层线性前馈网络对图像编码，每个神经元有 64 个输入。利用 8×8 的非重叠图像块训练网络。试验扫描图像 2000 次，学习率 $\eta=10^{-4}$。

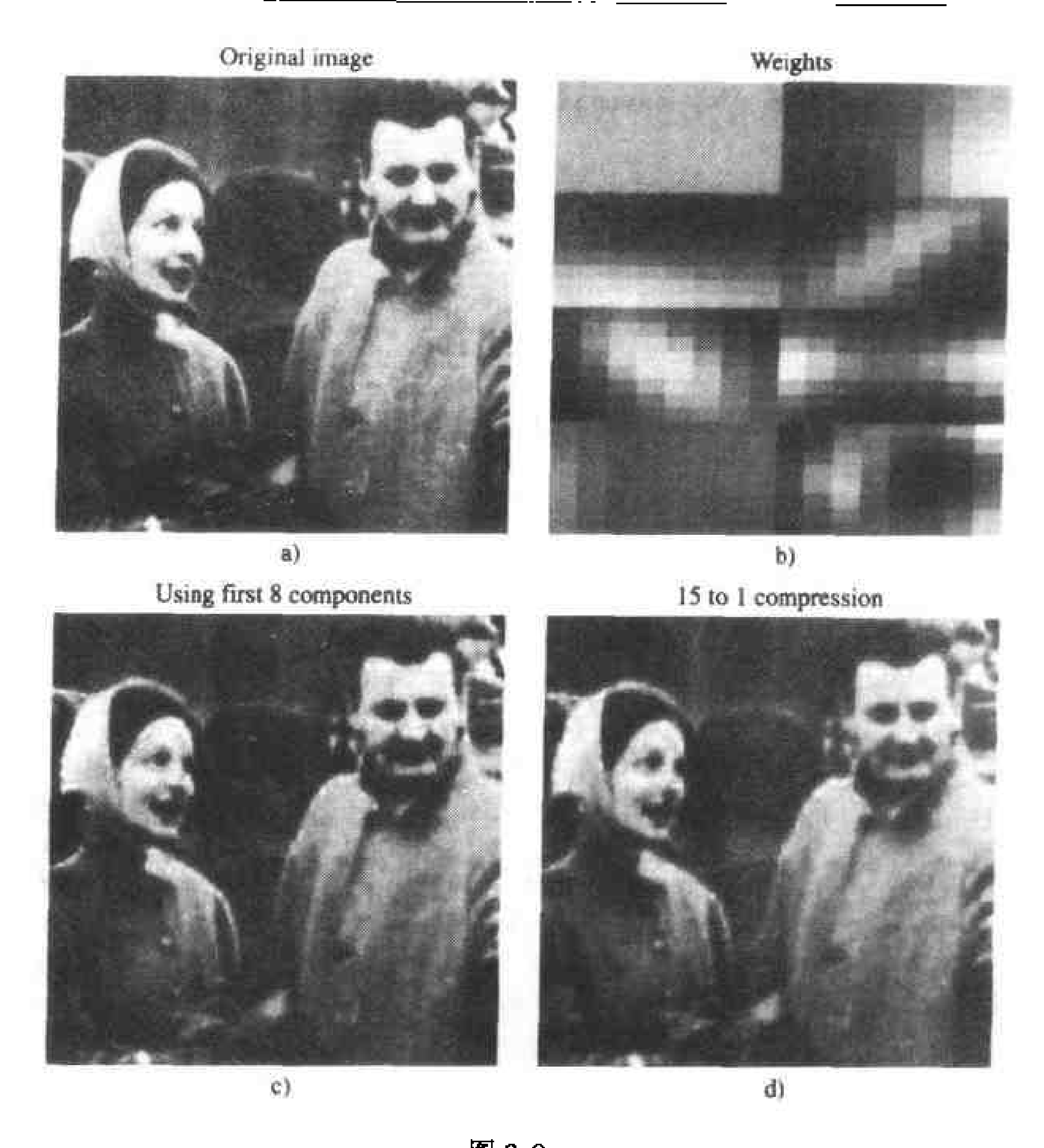
图 8-9b 显示的 8×8 的屏蔽(mask)表示网络学习所得的突触权值。8 个屏蔽中的每一个为与某个特定的神经元相关的一组权值。具体地，兴奋(正)的权值用白色显示，抑制(负)的权值用黑色表示，灰色表示权值为 0。在我们的表示法中，屏蔽表示广义 Hebb 算法收敛后的 64×8 突触权值矩阵 $\mathbf{W}^T$ 的列。

使用下面的步骤实现对图像编码：

- 图像的每个 8×8 块与图 8-9b 所示的 8 个屏蔽的每一个相乘，因此将产生 8 个系数作为图像编码；图 8-9c 显示没有量化的基于 8 个主分量的图像重建。
- 每个系数一律被量化为与该图像的系数方差的对数成正比的比特数。最大的 3 个屏蔽为每个 6 比特，其次的两个为每个 4 比特，再其次的两个为每个 3 比特，最小的一个为 2 比特。基于上述表示，需要 34 比特对每 8×8 的像素块编码，每个像素为 0.53 比特的数据率。

用量化系数重建图像，所有的屏蔽都用它们的量化系数加权，然后叠加重新构成的每块图像。以 15:1 的压缩率重建双亲图像如图 8-9d 所示。[419]

作为第一个图像的变化，下面我们对图 8-10a 所示的海洋景色图片应用广义 Hebb 算法。



a) b) c) d)

图 8-9

a)用于图像编码试验的双亲图像 b)8×8 的屏蔽表示由 GHA 学习的突触权值

c)利用 8 个无量化主分量所得的双亲图像重建 d)用量化的 15:1 压缩比的双亲图像重建

这幅图像强调纹理信息。图 8-10b 显示用前面描述的处理方式由网络学得的突触权值的 8×8 屏蔽图像；注意到它们和 8-9b 的屏蔽的区别。图 8-10c 显示没有量化的基于 8 个主分量重建的海洋图像。为了研究量化的影响，令前两个屏蔽的输出每个为 5 比特，第 3 个为 3 比特，剩下的 5 个每个为 2 比特。这样需要 23 比特为每个 8×8 像素块编码，每个像素块的比特率为 0.36 比特每像素。图 8-10d 显示量化后重建的海洋景色图像，使用自己的以刚才描述的方式量化的屏蔽。这幅图像的压缩比为 22:1。

420

为了测试广义 Hebb 算法的"泛化"性能，最后用图 8-9b 的屏蔽分解图 8-10a 所示的海洋景色图像，然后用与产生图 8-10d 所示重建图像一样的量化过程。这个图像重建结果如图 8-10e所示，压缩比与 8-10d 一样，也为 22:1。虽然在 8-10d 中的重建图像与在 8-10e 中的是惊人地一致，但可以看到图 8-10d 比 8-10e 更具有真实纹理信息而更少块状现象。产生这种情况的原因在于网络的权值。对双亲图像和海洋景色图像所完成的训练，它们的前 4 个突触权值很相似。然而，对双亲图像而言，后 4 个权值编码边缘信息，但在海洋景色图像中，这 4 个权值编码纹理信息。因此当用边缘型权值对海洋图像编码时，纹理数据在重建后是粗糙的，因此产生了块状现象。

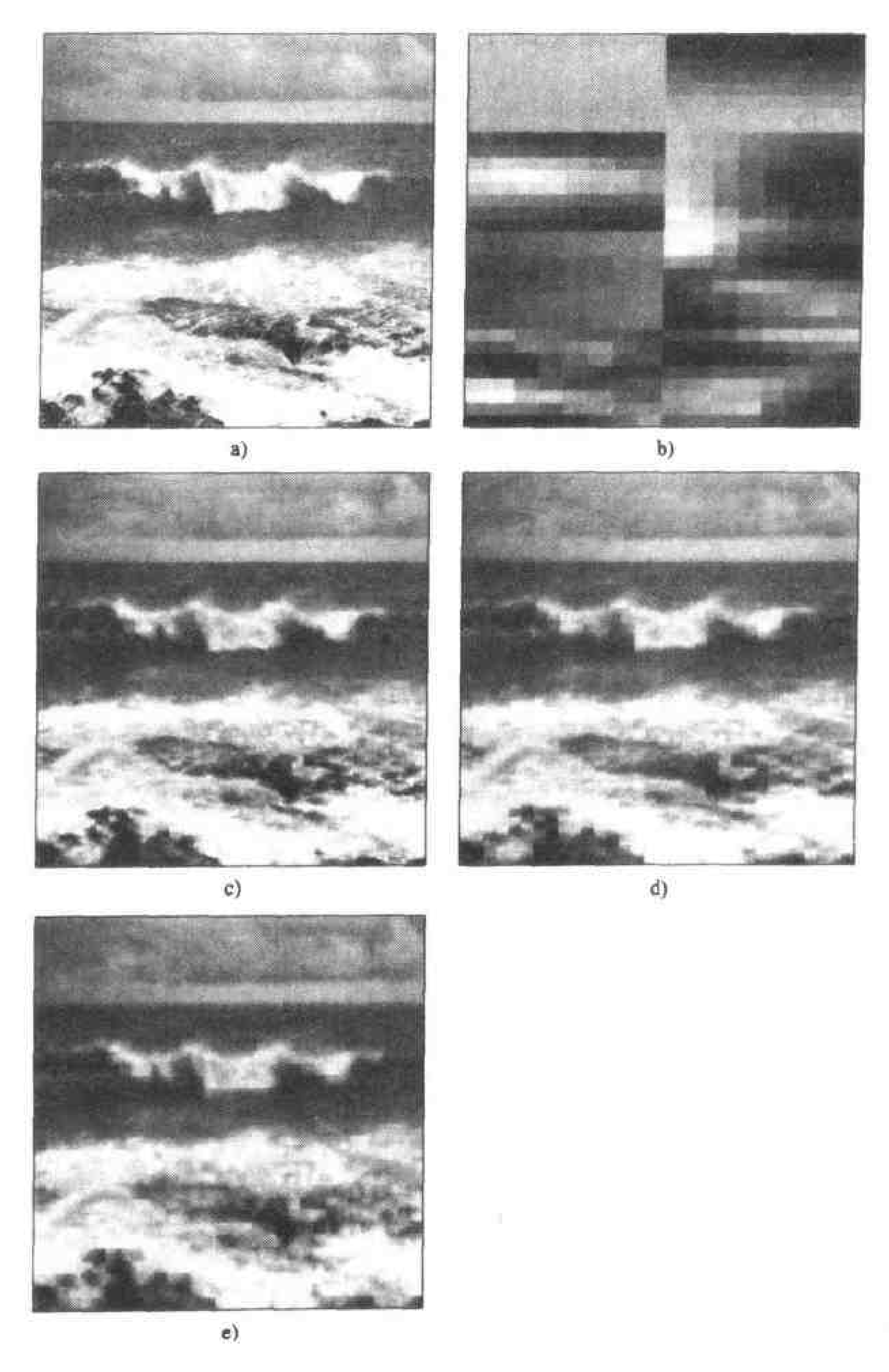


图 8-10

a)海洋景色图像 b)8×8 的屏蔽表示由应用于海洋景色的 GHA 学习到的突触权值 c)利用 8 个优势主分量重建的海洋景色图像 d)利用 b)中的屏蔽以 22:1 压缩比重建的海洋景色图像 e)利用图 8-9b 中的屏蔽编码以 22:1 压缩比量化重建的海洋景色图像

421

8.7 使用侧向抑制的自适应主分量分析

前一节描述的广义 Hebb 算法是基于排除使用前馈连接的主分量分析。在这一节我们讨论另一个称之为自适应主分量抽取(adaptive principal components extraction, APEX)的算法(Kung

and Diamantaras, 1990; Diamantaras and Kung, 1996)。APEX算法使用前馈连接和反馈连接[3]。其特点是如果给出前$(j-1)$个主分量，它可以用迭代方式计算第j个主分量。

用于导出APEX算法的网络模型如图8-11所示。和以前一样，输入向量$\mathbf{x}$为m维，其分量用$x_1, x_2, \cdots, x_m$表示。网络中每个神经元均为线性单元。如图8-11的描绘，网络中有两种突触连接方式：

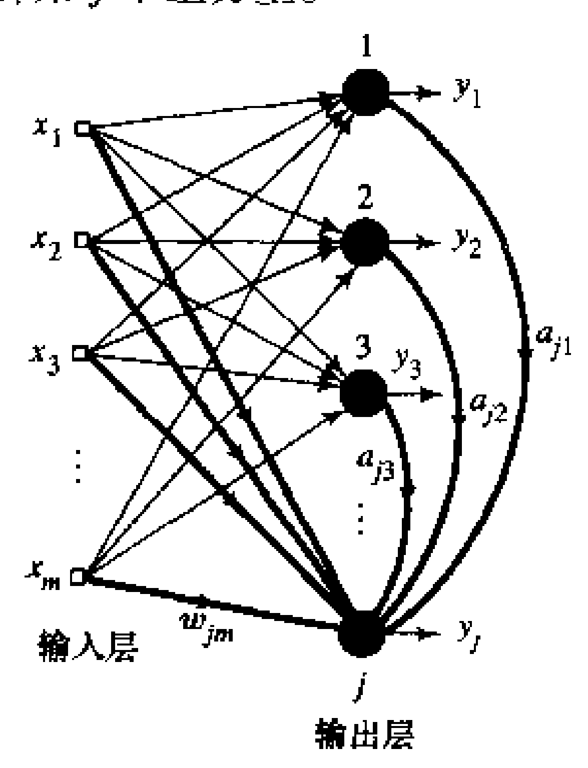


图8-11 用前馈连接和侧向连接的网络，用于导出APEX算法

- 前馈连接：由输入节点到神经元$1,2,\cdots,j$间的连接，$j<m$。我们特别感兴趣的是到神经元j的前馈连接权值向量；这些连接由前馈权值向量

$$\mathbf{w}_j = [w_{j1}(n), w_{j2}(n), \cdots, w_{jm}(n)]^T$$

表示。前馈连接按照Hebb学习规则运行；这种连接是兴奋性的，从而起到自增强作用。

[422]

- 侧向连接：从输出单个神经元$1,2,\cdots,j-1$到神经元j间的连接，对网络起反馈作用。这些连接由反馈突触权值向量

$$\mathbf{a}_j(n) = [a_{j1}(n), a_{j2}(n), \cdots, a_{j,j-1}(n)]^T$$

表示。侧向连接按反Hebb学习规则(anti-Hebb learning rule)运行，该规则对它们产生抑制作用。

在图8-11中，第j个神经元的前馈连接和反馈连接用粗线表示仅仅为了强调神经元j是研究的主题。

神经元j的输出$y_j(n)$为

$$y_j(n) = \mathbf{w}_j^T(n)\mathbf{x}(n) + \mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n) \tag{8.98}$$

其中$\mathbf{w}_j^T(n)\mathbf{x}(n)$由前馈连接产生，$\mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n)$由侧向连接产生。反馈信号向量$\mathbf{y}_{j-1}(n)$由神经元$1,2,\cdots,j-1$的输出定义：

$$\mathbf{y}_{j-1}(n) = [y_1(n), y_2(n), \cdots, y_{j-1}(n)]^T \tag{8.99}$$

假定输入信号$\mathbf{x}(n)$取自平稳随机过程，其相关矩阵$\mathbf{R}$具有不同的特征值并按递减顺序排列如下：

$$\lambda_1 > \lambda_2 > \cdots > \lambda_{j-1} > \lambda_j > \cdots > \lambda_m \tag{8.100}$$

进一步假设图8-11中网络的神经元$1,2,\cdots,j-1$已经收敛到相应的稳定条件，即

$$\mathbf{w}_k(0) = \mathbf{q}_k, \qquad k = 1,2,\cdots,j-1 \tag{8.101}$$

$$\mathbf{a}_k(0) = \mathbf{0}, \qquad k = 1,2,\cdots,j-1 \tag{8.102}$$

其中$\mathbf{q}_k$是与相关矩阵$\mathbf{R}$的第k个特征值相联系的特征向量，网络神经元j从时间步$n=0$时开始计算。我们可以利用式(8.98)、(8.99)、(8.101)和(8.102)写成

$$\mathbf{y}_{j-1}(n) = [\mathbf{q}_1^T\mathbf{x}(n), \mathbf{q}_2^T\mathbf{x}(n), \cdots, \mathbf{q}_{j-1}^T\mathbf{x}(n)] = \mathbf{Q}\mathbf{x}(n) \tag{8.103}$$

其中$\mathbf{Q}$是$(j-1)\times m$矩阵，由相关矩阵$\mathbf{R}$的$(j-1)$个最大的特征值$\lambda_1, \lambda_2, \cdots, \lambda_{j-1}$相联系的特征向量$\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_{j-1}$构成，即

[423]

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_{j-1}]^T \tag{8.104}$$

下面的任务是用图8-11中网络的神经元 j 计算输入向量的相关矩阵 $\mathbf{R}$ 的下一个最大特征值 λ_j 和它对应的特征向量 $\mathbf{q}_j$。

前馈突触权值 $\mathbf{w}_j(n)$ 和反馈突触权值 $\mathbf{a}_j(n)$ 的更新方程分别定义为

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta[y_j(n)\mathbf{x}(n) - y_j^2(n)\mathbf{w}_j(n)] \tag{8.105}$$

和

$$\mathbf{a}_j(n+1) = \mathbf{a}_j(n) - \eta[y_j(n)\mathbf{y}_{j-1}(n) + y_j^2(n)\mathbf{a}_j(n)] \tag{8.106}$$

其中 η 是学习率参数，假设两个更新方程中的 η 一样。式(8.106)右端的 $y_j(n)\mathbf{x}(n)$ 项代表Hebb学习，而项 $-y_j(n)\mathbf{y}_{j-1}(n)$ 代表反Hebb学习。剩下的项 $y_j^2(n)\mathbf{a}_j(n)$ 和 $-y_j^2(n)\mathbf{w}_j(n)$ 保证算法的稳定性。基本上，式(8.105)是式(8.40)所述的Oja学习规则的矢量形式，而(8.106)是新的，说明侧向连接的作用(Kung and Diamantaras, 1990; Diamantaras and Kung, 1996)。

可用归纳法证明图8-11神经网络的绝对稳定性如下：

- 首先，我们证明如果神经元 $1,2,\cdots,j-1$ 收敛于其稳定状态，那么神经元 j 将通过提取输入向量 $\mathbf{x}(n)$ 的相关矩阵 $\mathbf{R}$ 的第 j 个特征值 λ_j 及其对应的特征向量 $\mathbf{q}_j$ 而达到自身的稳定状态。
- 其次，认识到神经元1没有反馈连接，因此反馈权值向量 $\mathbf{a}_1$ 是0，我们可由归纳法完成这个证明。因此这个特殊的神经元运行实际上与Oja神经元的运行过程一样，由8.4节知道在一定条件下这个神经元绝对收敛。

因此仅仅需要注意第一点。

为了进一步处理，我们使用8.4节所作的基本假设，在图8-11所示网络中的神经元 j 的运行满足式(8.105)和(8.106)描述的条件下，我们得到下面的定理(Kung and Diamantaras, 1990; Diamantaras and Kung, 1996)：

若给定的学习率参数 η 足够小，使权值向量的调节进行缓慢，在极限时前馈连接的权值向量和神经元 j 的平均输出功率(方差)趋近于相关矩阵 $\mathbf{R}$ 的归一化特征向量 $\mathbf{q}_j$ 和对应的特征值 λ_j，分别表示为 [424]

$$\lim_{n\to\infty} \mathbf{w}_j(n) = \mathbf{q}_j$$

和

$$\lim_{n\to\infty} \sigma_j^2(n) = \lambda_j$$

其中 $\sigma_j^2(n) = E[y_j^2(N)]$，且 $\lambda_1 > \lambda_2 > \cdots > \lambda_j > \cdots > \lambda_m > 0$。换句话说，给定特征向量 $\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_{j-1}$，图8-11所示网络的神经元 j 计算出下一个神经元的最大特征值 λ_j 和对应的特征向量 $\mathbf{q}_j$。

为了证明这个定理，首先考虑式(8.105)。利用式(8.98)和(8.99)，并且认识到：

$$\mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n) = \mathbf{y}_{j-1}^T(n)\mathbf{a}_j(n)$$

可以改写式(8.105)如下：

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}_j(n) + \mathbf{x}(n)\mathbf{x}^T(n)\mathbf{Q}^T\mathbf{a}_j(n) - y_j^2(n)\mathbf{w}_j(n)] \tag{8.107}$$

其中 $\mathbf{Q}$ 由式(8.104)定义。在式(8.107)中项 $y_j^2(n)$ 没有改变，其原因后面将会明白。用8.4节的基本假设，对(8.107)两端应用统计期望算子可得

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta[\mathbf{R}\mathbf{w}_j(n) + \mathbf{R}\mathbf{Q}^T\mathbf{a}_j(n) - \sigma_j^2(n)\mathbf{w}_j(n)] \tag{8.108}$$

其中 $\mathbf{R}$ 是输入向量 $\mathbf{x}$ 的相关矩阵，$\sigma_j^2(n)$ 是神经元 j 的平均输出功率。令权值向量 $\mathbf{w}_j(n)$ 被

展开成相关矩阵 $\mathbf{R}$ 的正交特征向量集如下：

$$\mathbf{w}_j(n) = \sum_{k=1}^{m} \theta_{jk}(n)\mathbf{q}_k \tag{8.109}$$

其中 $\mathbf{q}_k$ 是矩阵 $\mathbf{R}$ 的 λ_k 对应的特征向量，$\theta_{jk}(n)$是展开式的时变系数。利用基本关系(参看式(8.14))

$$\mathbf{R}\mathbf{q}_k = \lambda_k \mathbf{q}_k$$

表示矩阵乘积 $\mathbf{R}\mathbf{w}_j(n)$如下：

$$\mathbf{R}\mathbf{w}_j(n) = \sum_{k=1}^{m} \theta_{jk}(n)\mathbf{R}\mathbf{q}_k = \sum_{k=1}^{m} \lambda_k \theta_{jk}(n)\mathbf{q}_k \tag{8.110}$$

类似地，用式(8.104)表示矩阵乘积 $\mathbf{R}\mathbf{Q}^T\mathbf{a}_j(n)$为

$$\mathbf{R}\mathbf{Q}^T\mathbf{a}_j(n) = \mathbf{R}[\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_{j-1}]\mathbf{a}_j(n)$$

425

$$= [\lambda_1\mathbf{q}_1, \lambda_2\mathbf{q}_2, \cdots, \lambda_{j-1}\mathbf{q}_{j-1}]\begin{bmatrix} a_{j1}(n) \\ a_{j2}(n) \\ \vdots \\ a_{j,j-1}(n) \end{bmatrix} = \sum_{k=1}^{j-1} \lambda_k a_{jk}(n)\mathbf{q}_k \tag{8.111}$$

因此，将式(8.109)、(8.110)和(8.111)代入式(8.108)并化简，得到(Kung and Diamantaras, 1990)

$$\sum_{k=1}^{m} \theta_{jk}(n+1)\mathbf{q}_k = \sum_{k=1}^{m} \{1 + \eta[\lambda_k - \sigma_j^2(n)]\}\theta_{jk}(n)\mathbf{q}_k + \eta \sum_{k=1}^{j-1} \lambda_k a_{jk}(n)\mathbf{q}_k \tag{8.112}$$

遵循上述类似的过程，可以将关于反馈权值向量 $\mathbf{a}_j(n)$的更新方程(8.106)变换成下述形式(参看习题 8.7)：

$$\mathbf{a}_j(n+1) = -\eta\lambda_k\theta_{jk}(n)\mathbf{1}_k + \{1 - \eta[\lambda_k + \sigma_j^2(n)]\}\mathbf{a}_j(n) \tag{8.113}$$

其中 $\mathbf{1}_k$ 是第 j 个元素为 1 而其他元素均为 0 的向量。下标 k 被限制在范围 $1 \le k \le j-1$ 内。

按 k 与 $j-1$ 的关系需考虑两种情况。情况 I 指 $1 \le k \le j-1$，适用于分析网络"已有的"主模式。情况 II 指 $j \le k \le m$，适用于分析"新的"主模式，而总的数量为 m，即输入向量 $\mathbf{x}(n)$的维数。

情况 I $1 \le k \le j-1$ 在这种情况下，从式(8.112)和(8.113)分别推出关于 $\mathbf{q}_k$ 的系数 $\theta_{jk}(n)$的更新方程以及反馈权值向量 $a_{jk}(n)$的更新方程

$$\theta_{jk}(n+1) = \eta\lambda_k a_{jk}(n) + \{1 + \eta[\lambda_k - \sigma_j^2(n)]\}\theta_{jk}(n) \tag{8.114}$$

426 和

$$a_{jk}(n+1) = -\eta\lambda_k\theta_{jk}(n) + \{1 - \eta[\lambda_k + \sigma_j^2(n)]\}a_{jk}(n) \tag{8.115}$$

图 8-12 给出式(8.114)和(8.115)所描述的信号流图。

用矩阵形式重写式(8.114)和(8.115)如下：

$$\begin{bmatrix} \theta_{jk}(n+1) \\ a_{jk}(n+1) \end{bmatrix} = \begin{bmatrix} 1 + \eta[\lambda_k - \sigma_j^2(n)] & \eta\lambda_k \\ -\eta\lambda_k & 1 - \eta[\lambda_k + \sigma_j^2(n)] \end{bmatrix} \begin{bmatrix} \theta_{jk}(n) \\ a_{jk}(n) \end{bmatrix} \tag{8.116}$$

式(8.116)描述的系统矩阵在

$$\rho_{jk} = [1 - \eta\sigma_j^2(n)]^2 \tag{8.117}$$

时具有重特征值。由式(8.117)可得到下面的重要结论：

1. 式(8.117)中系统矩阵的重特征值ρ_{jk}不依赖于相关矩阵$\mathbf{R}$的特征值λ_k，$k=1,2,\cdots,j-1$。

2. 对于所有的k，ρ_{jk}只取决于学习率参数η和神经元j的平方输出功率σ_j^2。只要学习率参数η为足够小，则它为小于1的正数。

假如$\rho_{jk}<1$，式(8.109)中的系数$\theta_{jk}(n)$和反馈权值$a_{jk}(n)$对所有的k以同样的速度趋向于0，因为网络的主模式具有同样的特征值(Kung and Diamantaras,1990;Diamantaras and Kung,1996)。这个结果基于这样的性质，即特征向量的正交性不依赖于特征值。换句话说，式(8.109)中$\mathbf{w}_j(n)$对相关矩阵$\mathbf{R}$的正交特征向量集的展开式与特征值$\lambda_1,\lambda_2,\cdots,\lambda_{j-1}$的选择是无关的，式(8.109)对式(8.117)的结果是基本的。

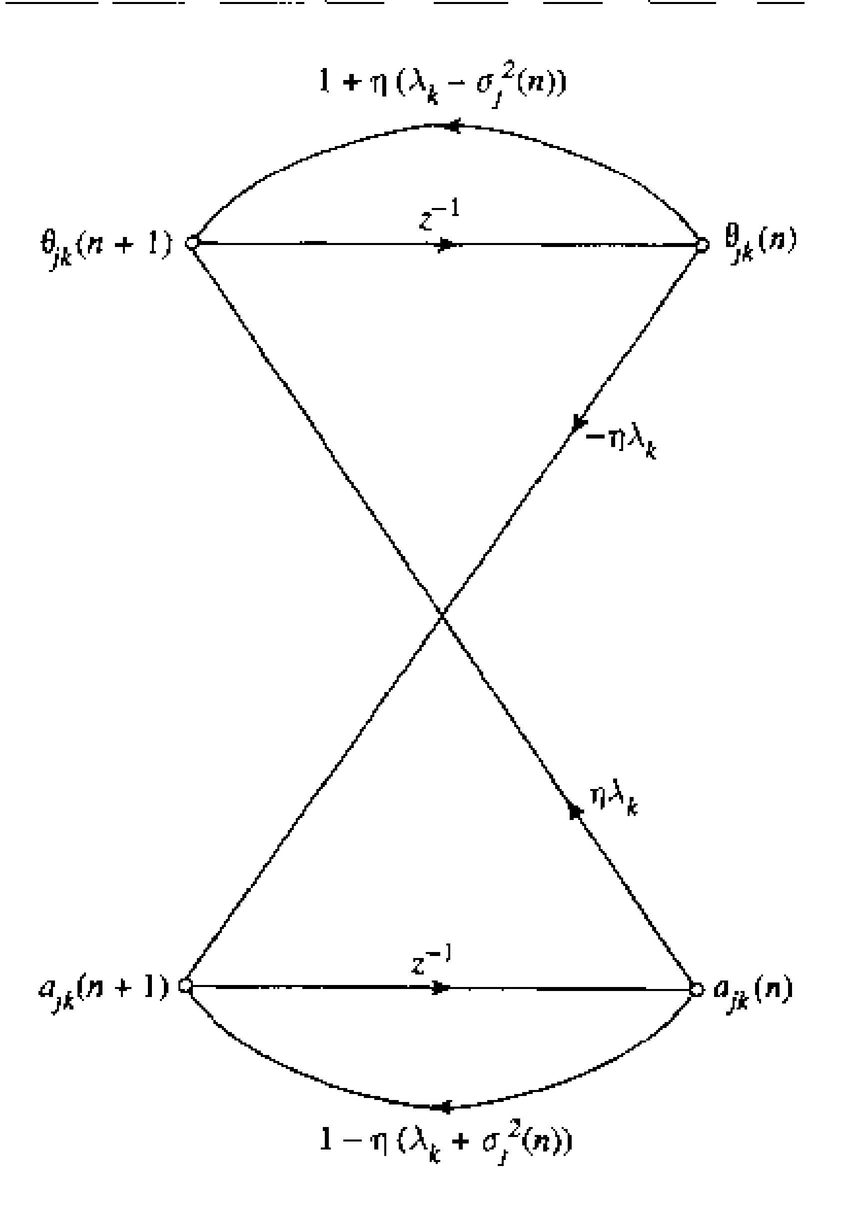


图 8-12 式(8.114)和(8.115)的信号流图表示

情况 II $j\leqslant k\leqslant m$ 在第二种情况下，反馈权值$a_{jk}(n)$对网络模式(mode)无影响，即

$$a_{jk}(n)=0 \qquad 对于\ j\leqslant k\leqslant m \tag{8.118}$$

因此，对每个主模式$k\geqslant j$我们有下面很简单的等式：

$$\theta_{jk}(n+1)=\{1+\eta[\lambda_k-\sigma_j^2(n)]\}\theta_{jk}(n) \tag{8.119}$$

这直接由式(8.112)和(8.118)可得。根据情况I，对$k=1,2,\cdots,j-1$，$\theta_{jk}(n)$和$a_{jk}(n)$都收敛于0。用随机变量$Y_j(n)$表示神经元j的输出，平均输出功率可以表示如下：

$$\sigma_j^2(n)=E[Y_j^2(n)]=\sum_{k=j}^{m}\lambda_k\theta_{jk}^2(n) \tag{8.120}$$

[427]

其中第二个等式使用了下列关系：

$$\mathbf{q}_k^T\mathbf{R}\mathbf{q}_l=\begin{cases}\lambda_k, & l=k\\ 0, & 其他\end{cases}$$

因此式(8.119)不可能发散，因为无论$\theta_{jk}(n)$变得多大，只要$\sigma_j^2(n)>\lambda_k$，则$1+\eta[\lambda_k-\sigma_j^2(n)]$变成小于1，在这种情况下，$\theta_{jk}(n)$的幅值将减小。令算法用初始值$\theta_{jj}(0)\neq 0$，同时定义

$$r_{jk}(n)=\frac{\theta_{jk}(n)}{\theta_{jj}(n)}, \qquad k=j+1,\cdots,m \tag{8.121}$$

可以用式(8.119)写为

$$r_{jk}(n+1)=\frac{1+\eta[\lambda_k-\sigma_j^2(n)]}{1+\eta[\lambda_j-\sigma_j^2(n)]}r_{jk}(n) \tag{8.122}$$

相关矩阵的特征值按降序排列，

$$\lambda_1>\lambda_2>\cdots>\lambda_k>\cdots>\lambda_j>\cdots>\lambda_m$$

由此推出

$$\frac{\theta_{jk}(n)}{\theta_{jj}(n)} < 1 \quad \text{对于所有 } n \text{ 和 } k = j+1,\cdots,m \tag{8.123}$$

此外，我们注意从式(8.119)和(8.120)可得 $\theta_{jj}(n+1)$有界，因此

$$r_{jk}(n) \to 0 \quad \text{当 } n \to \infty \text{ 时} \qquad \text{对于 } k = j+1,\cdots,m \tag{8.124}$$

同样地，按照式(8.121)的定义，我们可得

$$\theta_{jk}(n) \to 0 \quad \text{当 } n \to \infty \text{ 时} \quad \text{对于 } k = j+1,\cdots,m \tag{8.125}$$

在这个条件下，式(8.120)简化为

$$\sigma_j^2(n) = \lambda_j \theta_{jj}^2(n) \tag{8.126}$$

所以式(8.119)对 $k=j$ 变为

$$\theta_{jj}(n+1) = \{1 + \eta\lambda_j[1 - \theta_{jj}(n)]\}\theta_{jj}(n) \tag{8.127}$$

从上式可立即推出

$$\theta_{jj}(n) \to 1 \quad \text{当 } n \to \infty \text{ 时} \tag{8.128}$$

这个极限条件和式(8.125)的极限条件有两个方面的含义：

1. 从式(8.126)我们有

$$\sigma_j^2(n) \to \lambda_j \quad \text{当 } n \to \infty \text{ 时} \tag{8.129}$$

2. 从式(8.109)我们有

[428]

$$\mathbf{w}_j(n) \to \mathbf{q}_j \quad \text{当 } n \to \infty \text{ 时} \tag{8.130}$$

换句话说，当迭代数目 n 趋于无穷大时图 8-11 的神经网络模型抽出输入向量 $\mathbf{x}(n)$的相关矩阵 $\mathbf{R}$ 的第 j 个特征值和对应的特征向量。这时自然假定网络的神经元$1,2,\cdots,j-1$ 都已经收敛于相关矩阵 $\mathbf{R}$ 的对应特征值和特征向量。

这里描述的 APEX 算法的前提为，在神经元 j 开始作用前，神经元$1,2,\cdots,j-1$ 都已经收敛。这是为了简化对算法运行的解释。实际上，APEX 算法中的神经元是同时收敛的[4]。

学习率

在式(8.105)和(8.106)中描述的 APEX 算法中，更新前馈权值向量 $\mathbf{w}_j(n)$和反馈权值向量 $\mathbf{a}_j(n)$的学习率参数 η 是相同的。通过置重特征值 ρ_{jk} 为 0，式(8.117)可被用来为每个神经元 j 定义学习率 η 的最佳值。在这个情况下，有

$$\eta_{j,\text{opt}}(n) = \frac{1}{\sigma_j^2(n)} \tag{8.131}$$

其中 $\sigma_j^2(n)$是神经元 j 的平均输出功率。但是，更实际的建议是置(Kung and Diamantaras, 1990; Diamantaras and Kung, 1996)

$$\eta_j = \frac{1}{\lambda_{j-1}} \tag{8.132}$$

因为 $\lambda_{j-1} > \lambda_j$ 且当$n \to \infty$时 $\sigma_j^2(n) > \lambda_j$，因此对学习率参数 η 产生过低的值。注意特征值 λ_{j-1} 由神经元 $j-1$ 计算得到，因此对神经元 j 的前馈和反馈权值的更新都是可用的。

APEX 算法小结

1. 在 $n=1$ 时，对前馈权值向量 $\mathbf{w}_j$ 和反馈权值向量 $\mathbf{a}_j$ 赋于小的随机数作为初值，其中

$j=1,2,\cdots,m$。设定学习率参数 η 为小的正数。

2. 置 $j=1$，对 $n=1,2,\cdots$ 计算

$$y_1(n)=\mathbf{w}_1^T(n)\mathbf{x}(n)$$
$$\mathbf{w}_1(n+1)=\mathbf{w}_1(n)+\eta[y_1(n)\mathbf{x}(n)-y_1^2(n)\mathbf{w}_1(n)]$$

其中 $\mathbf{x}(n)$为输入向量。对于很大的 n，有 $\mathbf{w}_1(n)\to\mathbf{q}_1$，$\mathbf{q}_1$ 为 $\mathbf{x}(n)$的相关矩阵的最大特征值 λ_1 对应的特征向量。

3. 置 $j=2$，对 $n=1,2,\cdots$ 计算

$$\mathbf{y}_{j-1}(n)=[y_1(n),y_2(n),\cdots,y_{j-1}(n)]^T$$
$$y_j(n)=\mathbf{w}_j^T(n)\mathbf{x}(n)+\mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n)$$
$$\mathbf{w}_j(n+1)=\mathbf{w}_j(n)+\eta[y_j(n)\mathbf{x}(n)-y_j^2(n)\mathbf{w}_j(n)]$$
$$\mathbf{a}_j(n+1)=\mathbf{a}_j(n)-\eta[y_j(n)\mathbf{y}_{j-1}(n)+y_j^2(n)\mathbf{a}_j(n)]$$

4. 对于增加 1，返回第 3 步，并继续直到 $j=m$，其中 m 是期望的主分量的数量。(注意 $j=1$ 对最大特征值相关的特征向量，在第 2 步受到处理)对于很大的 n，我们有 $\mathbf{w}_j(n)\to\mathbf{q}_j$，$\mathbf{a}_j(n)\to\mathbf{0}$，其中 $\mathbf{q}_j$ 是 $\mathbf{x}(n)$的相关矩阵的第 j 个特征值对应的特征向量。 429

8.8 两类 PCA 算法

除了 8.5 节讨论的广义 Hebb 算法(GHA)和 8.7 节讨论的 APEX 算法外，在文献[5]中还报导了几种其他的主分量分析算法。神经网络中使用的各种主分量分析(PCA)可分为两类：重估计(reestimation)算法和去相关(decorrelating)算法。

按照这个分类，GHA 是重估计算法，因为式(8.87)和(8.88)可重写为等价的形式

$$\mathbf{w}_j(n+1)=\mathbf{w}_j(n)+\eta y_j(n)[\mathbf{x}(n)-\hat{\mathbf{x}}_j(n)] \tag{8.133}$$

其中重估计算子 $\hat{\mathbf{x}}_j(n)$定义为

$$\hat{\mathbf{x}}_j(n)=\sum_{k=1}^{j}\mathbf{w}_k(n)y_k(n) \tag{8.134}$$

在重估计算法中神经网络只有前馈连接，按 Hebb 方式修改它的强度(权值)。通过在学习过程涉及数据集之前先从输入中减掉前几个主分量的估计值，强迫网络的后继输出学习不同主分量。

相反，APEX 算法是去相关算法。在这种算法中网络具有前馈和反馈连接，前馈连接的强度遵守 Hebb 规则，而反馈连接的强度遵守反 Hebb 规则。网络的后继输出通过去相关作用来强迫网络响应不同的主分量。

主子空间

在仅需要主子空间(即主分量对应的空间)的情况下，我们用一种对称模型替代 GHA 算法中的重估计算子 $\hat{\mathbf{x}}_j(n)$：

$$\hat{\mathbf{x}}_j(n)=\sum_{k=1}^{l}\mathbf{w}_k(n)y_k(n)\qquad 对于所有\ l \tag{8.135}$$

在式(8.133)和(8.135)定义的对称模型中，网络收敛于一组可生成主子空间的输出，而不是主分量本身。收敛时权值向量彼此正交，如在 GHA 算法中一样。这里描述的主子空间可被

认为是由式(8.46)定义的经典 Oja 规则的一种推广。

8.9 计算的集中式方法和自适应方法

讨论主分量分析时不考虑问题的计算方面是不完整的。在本节将讨论两个主分量计算的基本方法：集中式方法和自适应方法。在8.3节描述的特征分解和相关的奇异值分解方法属于集中式类。另一方面，在8.5节和8.7节讨论的 GHA 算法和 APEX 算法属于自适应类。

430

在理论上，如8.3节的描述特征分解方法是基于输入随机向量 $\mathbf{X}(n)$ 的相关矩阵 $\mathbf{R}$ 的总体平均。实际上，我们使用 $\mathbf{R}$ 的估计值。令 $\{\mathbf{x}(n)\}_{n=1}^{N}$ 表示随机向量 $\mathbf{X}(n)$ 在均匀间隔的离散时刻的一组 N 次实现。给定这样一组观察，我们可以用样本均值作为相关矩阵的估计：

$$\hat{\mathbf{R}}(N) = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}(n)\mathbf{x}^{T}(n) \tag{8.136}$$

只要用 $\mathbf{X}(n)$ 表示的输入环境向量为各态历经的，当样本大小 N 趋于无穷大时，样本均值 $\hat{\mathbf{R}}(N)$ 趋于 $\mathbf{R}$。在这个基础上，可以对样本均值 $\hat{\mathbf{R}}(N)$ 使用特征分解过程，从而在式(8.22)用 $\hat{\mathbf{R}}(N)$ 替代 $\mathbf{R}$，由此计算出它的特征值和对应的特征向量。

然而，从数值的角度看，更好的方法是直接利用数据矩阵进行奇异值分解(singular value decomposition, SVD)。对一组观察值 $\{\mathbf{x}(n)\}_{n=1}^{N}$，数据矩阵为

$$\mathbf{A} = [\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(N)] \tag{8.137}$$

除了比例因子 $1/N$ 外，容易看出相关矩阵 $\mathbf{R}$ 的估计 $\hat{\mathbf{R}}(N)$ 与矩阵乘积 $\mathbf{A}\mathbf{A}^{T}$ 完全相同。按照第5章讨论的奇异值分解定理，数据矩阵 $\mathbf{A}(n)$ 可以分解如下(Golub and Van Loan, 1996)：

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{T} \tag{8.138}$$

其中 $\mathbf{U}$ 和 $\mathbf{V}$ 是正交矩阵，这意味着

$$\mathbf{U}^{-1} = \mathbf{U}^{T} \tag{8.139}$$

和

$$\mathbf{V}^{-1} = \mathbf{V}^{T} \tag{8.140}$$

至于矩阵 $\mathbf{\Sigma}$，具有下面的结构形式：

431

$$\mathbf{\Sigma} = \left[\begin{array}{c|c} \begin{bmatrix} \sigma_1 & & & \mathbf{0} \\ & \sigma_2 & & \\ & & \ddots & \\ \mathbf{0} & & & \sigma_k \end{bmatrix} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] \tag{8.141}$$

其中 $k \leqslant m$，m 是观察向量 $\mathbf{x}(n)$ 的维数。实数 $\sigma_1, \sigma_2, \cdots, \sigma_k$ 称为数据矩阵 $\mathbf{A}$ 的奇异值。相应地，正交矩阵 $\mathbf{U}$ 的列称为左奇异向量，而正交矩阵 $\mathbf{V}$ 的列称为右奇异向量。数据矩阵 $\mathbf{A}$ 的奇异值分解与相关矩阵的估计 $\hat{\mathbf{R}}(N)$ 的特征值分解有下面的关系：

- 除了比例因子 $1/N^{1/2}$ 外，数据矩阵 $\mathbf{A}$ 的特征值是估计 $\hat{\mathbf{R}}(N)$ 的特征值的平方根。
- $\mathbf{A}$ 的左奇异向量是估计 $\hat{\mathbf{R}}(N)$ 的特征向量。

现在，我们可以看出奇异值分解比特征值分解具有的数值优点。对于给定计算精度，奇异值分解过程需要的数值精度为特征值分解的一半。此外，在计算机上用于实现奇异值分解的过程已有许多算法和高精度的定制程序可资利用(Golub and Van Loan, 1996; Haykin 1996)。然而，在实际中，存储需求限制这些程序使用的样本量不可能太大。

下面转到另一类自适应方法，这些方法可以对任意大的样本大小 N 工作。对所有的实

际问题，对 N 均没有限制。基于 Hebb 规则的神经网络是自适应方法的例子，它操作的思想来源于神经生物学。这类方法对存储的要求相对适中，因为特征值和特征向量的中间值不需存储。自适应算法的另一个诱人的特征是在非平稳环境中，与集中式方法相比，它具有以最优解和较低代价跟踪缓慢变化的固有能力。然而，随机逼近型自适应算法的主要缺点是收敛速度相当慢，这一点和经典的集中式技术比较处于不利地位；对大型的平稳问题尤其如此，即使是在并行神经网络硬件上实现自适应方法(Kotilainen,1993)。

8.10 核主分量分析

到目前为止本章讨论的 PCA 形式涉及到在输入(数据)空间上的计算。现在我们考虑另一种形式的 PCA，计算在特征空间上进行，它和输入空间是非线性的关系。我们打算使用的特征空间是依据 Mercer 定理的内积核定义的；内积核的概念在第 6 章的支持向量机中讨论。基于核的主分量分析思想归功于 Schölkopf et al.(1998)。

由于输入空间和特征空间的非线性关系，核 PCA 是非线性的。然而，并不像其他形式的非线性 PCA[6]，核 PCA 的实现依赖于线性代数。因此我们可以将核 PCA 看作是一般 PCA 的自然扩展。

432

令向量 $\boldsymbol{\varphi}(\mathbf{x}_j)$ 表示输入向量 $\mathbf{x}_j$ 在非线性映射：$\boldsymbol{\varphi}$: $\mathbb{R}^{m_0}\rightarrow\mathbb{R}^{m_1}$ 定义特征空间中导出的像，其中 m_0 是输入空间的维数，m_1 是特征空间的维数。给定一组样本 $\{\mathbf{x}_i\}_{i=1}^N$，我们有一组相应的特征向量 $\{\boldsymbol{\varphi}(\mathbf{x}_i)\}_{i=1}^N$。因此我们可以在特征空间定义由 $\tilde{\mathbf{R}}$ 表示的 $m_1\times m_1$ 相关矩阵如下：

$$\tilde{\mathbf{R}}=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\varphi}(\mathbf{x}_i)\boldsymbol{\varphi}^T(\mathbf{x}_i) \tag{8.142}$$

如同普通的 PCA，我们首先要做的就是确保特征向量 $\{\boldsymbol{\varphi}(\mathbf{x}_i)\}_{i=1}^N$ 的集合具有零均值：

$$\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\varphi}(\mathbf{x}_i)=\mathbf{0}$$

在特征空间上满足这个条件比在输入空间上更加困难；在习题 8.10 中我们描述一个过程来满足这个要求。假设特征向量已经聚集于中心，则可以在目前情况下改变式(8.14)，写成

$$\tilde{\mathbf{R}}\tilde{\mathbf{q}}=\tilde{\lambda}\tilde{\mathbf{q}} \tag{8.143}$$

其中 $\tilde{\lambda}$ 为 $\tilde{\mathbf{R}}$ 的特征值，$\tilde{\mathbf{q}}$ 为对应的特征向量。我们注意对 $\tilde{\lambda}\neq 0$ 满足式(8.143)的所有特征向量，落在特征向量 $\{\boldsymbol{\varphi}(\mathbf{x}_j)\}_{j=1}^N$ 集合生成的空间中。因此存在一组相应的系数 $\{\alpha_j\}_{j=1}^N$，用它们可写成

$$\tilde{\mathbf{q}}=\sum_{j=1}^{N}\alpha_j\boldsymbol{\varphi}(\mathbf{x}_j) \tag{8.144}$$

由此将式(8.142)和(8.144)代入式(8.143)得到

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_j\boldsymbol{\varphi}(\mathbf{x}_i)K(\mathbf{x}_i,\mathbf{x}_j)=N\tilde{\lambda}\sum_{j=1}^{N}\alpha_j\boldsymbol{\varphi}(\mathbf{x}_j) \tag{8.145}$$

其中 $K(\mathbf{x}_i,\mathbf{x}_j)$ 是内积核，通过特征向量由下式定义：

$$K(\mathbf{x}_i,\mathbf{x}_j)=\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j) \tag{8.146}$$

我们需要进一步计算式(8.145)使得完全用内积核来表示此关系。在式(8.145)等号的两边左乘以转置向量 $\boldsymbol{\varphi}^T(\mathbf{x}_k)$ 得

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_j K(\mathbf{x}_k,\mathbf{x}_i)K(\mathbf{x}_i,\mathbf{x}_j) = N\tilde{\lambda}\sum_{j=1}^{N}\alpha_j K(\mathbf{x}_k,\mathbf{x}_j), k = 1,2,\cdots,N \tag{8.147}$$

其中 $K(\mathbf{x}_k,\mathbf{x}_i)$，$K(\mathbf{x}_k,\mathbf{x}_j)$由式(8.146)定义。

现在引入下面两个矩阵定义：

- $N\times N$ 矩阵 $\mathbf{K}$，称为核矩阵，它的第 ij 个元素为内积核 $K(\mathbf{x}_i,\mathbf{x}_j)$
- $N\times 1$ 向量 $\boldsymbol{\alpha}$，第 j 个元素为参数 α_j

因此，可以将式(8.147)写成紧凑的矩阵形式

$$\mathbf{K}^2\boldsymbol{\alpha} = N\tilde{\lambda}\mathbf{K}\boldsymbol{\alpha} \tag{8.148}$$

其中矩阵的平方 $\mathbf{K}^2$ 表示 $\mathbf{K}$ 自身相乘。因为式(8.148)两端均有 $\mathbf{K}$，特征值问题感兴趣的全部解同样可用为更简单的特征值问题表示：

[433]

$$\mathbf{K}\boldsymbol{\alpha} = N\tilde{\lambda}\boldsymbol{\alpha} \tag{8.149}$$

令$\lambda_1\geq\lambda_2\geq,\cdots,\geq\lambda_N$ 表示核矩阵 $\mathbf{K}$ 的特征值，即

$$\lambda_j = N\tilde{\lambda}_j, \qquad j = 1,2,\cdots,N \tag{8.150}$$

其中 $\tilde{\lambda}_j$ 是相关矩阵 $\tilde{\mathbf{R}}$ 的第 j 个特征值。从而式(8.149)变成标准形式

$$\mathbf{K}\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha} \tag{8.151}$$

其中系数向量 $\boldsymbol{\alpha}$ 起到核矩阵 $\mathbf{K}$ 的特征值 λ 的对应特征向量的作用。系数向量 $\boldsymbol{\alpha}$ 是归一化的，因为要求将相关矩阵 $\tilde{\mathbf{R}}$ 的特征向量 $\tilde{\mathbf{q}}$ 归一化为单位长度，即

$$\tilde{\mathbf{q}}_k^T\tilde{\mathbf{q}}_k = 1 \qquad 对\ k = 1,2,\cdots,p \tag{8.152}$$

此处假设特征值为降序排列，λ_p 为核矩阵 $\mathbf{K}$ 的特征值的最小非零值。利用式(8.144)和(8.151)我们可以得到式(8.152)等价的归一化条件：

$$\boldsymbol{\alpha}_k^T\boldsymbol{\alpha}_k = \frac{1}{\lambda_k}, k = 1,2,\cdots,p \tag{8.153}$$

为了抽出主分量，需要计算特征向量 $\tilde{\mathbf{q}}_k$ 在特征空间上的投影如下：

[434]

$$\tilde{\mathbf{q}}_k^T\boldsymbol{\varphi}(\mathbf{x}) = \sum_{j=1}^{N}\alpha_{k,j}\boldsymbol{\varphi}^T(\mathbf{x}_j)\boldsymbol{\varphi}(\mathbf{x}) = \sum_{j=1}^{N}\alpha_{k,j}K(\mathbf{x}_j,\mathbf{x}), k = 1,2,\cdots,p \tag{8.154}$$

其中向量 $\mathbf{x}$ 是“测试”点，$\alpha_{k,j}$是矩阵 $\mathbf{K}$ 第 k 个特征值对应的特征向量 $\boldsymbol{\alpha}_k$ 的第j 个系数。式(8.154)的投影定义在 m_1 维特征空间中的非线性主分量(nonlinear principal component)。

图 8-13 说明核 PCA 的基本思想，其中特征空间经过变换 $\boldsymbol{\varphi}(\mathbf{x})$和输入空间是非线性相关的。图中的 a 和 b 部分分别称为输入空间和特征空间。图 8-13b 中的轮廓线表示在主特征向量上的投影为常数的线，特征向量用虚线箭头表示。在此图中，假设变换 $\boldsymbol{\varphi}(\mathbf{x})$用下面的方式选择：在特征空间中数据点诱导的像聚集在特征向量沿线。图 8-13a 显示输入空间上对应特征空间的线性等值线的非线性等值线。注意我们有意没有在输入空间上画特征向量的原像，因为它甚至可能不存在(Schölkopf et al.,1998)。

按照 Mercer 定理定义的内积核，我们在 m_1 维特征空间上执行普通的 PCA，维数 m_1 是设计参数。8.3 节描述的普通 PCA 的所有性质对核 PCA 均适用。尤其，核 PCA 在特征空间上是线性的，但在输入空间上是非线性的。因此，所有可用普通 PCA 进行特征提取和数据压缩的领域，进行非线性扩展 PCA 也有意义。

在第 6 章我们提出了三个构造内积核的方法，它们是基于利用多项式、径向基函数和双

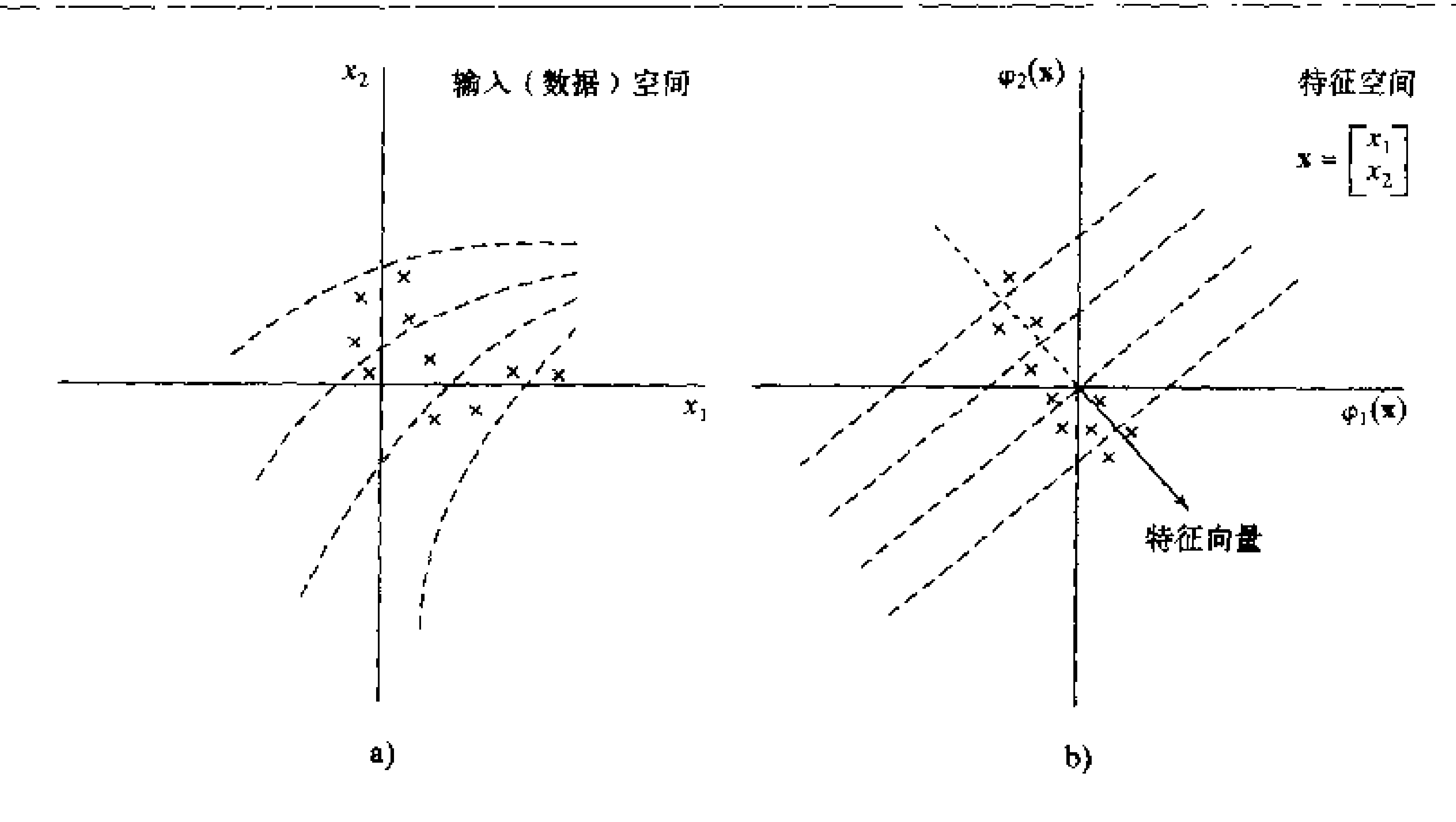


图 8-13　核 PCA 图例

a)二维输入空间，显示一组数据点　b)二维特征空间，显示数据点在一个主特征向量附近聚集的诱导像。在 b)中均匀排列的虚线表示在特征向量上投影为常数的等值线；它们在输入空间中的对应等值线是非线性的

曲函数；参见表 6-1。对给定的任务，怎么样选择最适合的核(即恰当的特征空间)是一个有待解决的问题(Schölkopf,1997)。

核主分量分析小结

1. 给定训练样本$\{\mathbf{x}_i\}_{i=1}^N$，计算 $N\times N$ 核矩阵 $\mathbf{K}=\{K(\mathbf{x}_i,\mathbf{x}_j)\}$，其中

$$K(\mathbf{x}_i,\mathbf{x}_j)=\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j)$$

2. 解释特征值问题：

$$\mathbf{K}\boldsymbol{\alpha}=\lambda\boldsymbol{\alpha}$$

其中 λ 为 $\mathbf{K}$ 的特征值，$\boldsymbol{\alpha}$ 为对应的特征向量。

3. 归一化所计算的特征值，这要求

$$\boldsymbol{\alpha}_k^T\boldsymbol{\alpha}_k=\frac{1}{\lambda_k},\qquad k=1,2,\cdots,p$$

其中 λ_p 是矩阵 $\mathbf{K}$ 最小的非零特征值，假设特征值是按降序排列的。

4. 为了抽取测试点 $\mathbf{x}$ 的主分量，计算投影

$$a_k=\tilde{\mathbf{q}}_k^T\boldsymbol{\varphi}(\mathbf{x})=\sum_{j=1}^N\alpha_{k,j}K(\mathbf{x}_j,\mathbf{x}),\qquad k=1,2,\cdots,p$$

其中 $\alpha_{k,j}$是特征向量 $\boldsymbol{\alpha}_k$ 的第j个元素。

435

例 8.3　为了对核 PCA 的运行有一个直观的了解，图 8-14 显示 Schölkopf et al.(1998)描述的一个简单的实验结果。二维数据由分量 x_1 和 x_2 组成，在这个试验中用下述方法产生：x_1 的值在区间[-1,1]均匀分布，x_2 的值与 x_1 的非线性相关，由

$$x_2=x_1^2+v$$

确定，其中 v 是加性 Gauss 白噪声，均值为 0，方差为 0.04。

图 8-14 所示的 PCA 的结果是用核多项式

$$K(\mathbf{x},\mathbf{x}_i) = (\mathbf{x}^T\mathbf{x}_i)^d, \qquad d = 1,2,3,4$$

得到的，其中 $d=1$ 对应线性 PCA，$d=2,3,4$ 对应于核 PCA。线性 PCA 如图 8-14 左面所示，因为输入空间为二维，仅产生两个特征向量。相反，核 PCA 允许抽出高阶分量，结果如图 8-14 中的 2、3、4 列所示，分别与 $d=2,3,4$ 对应。图中每部分显示的等值线(在线性 PCA 情形时除去零特征值)表示常数主值(即在与特征值相关联的特征向量上的投影为常数)。

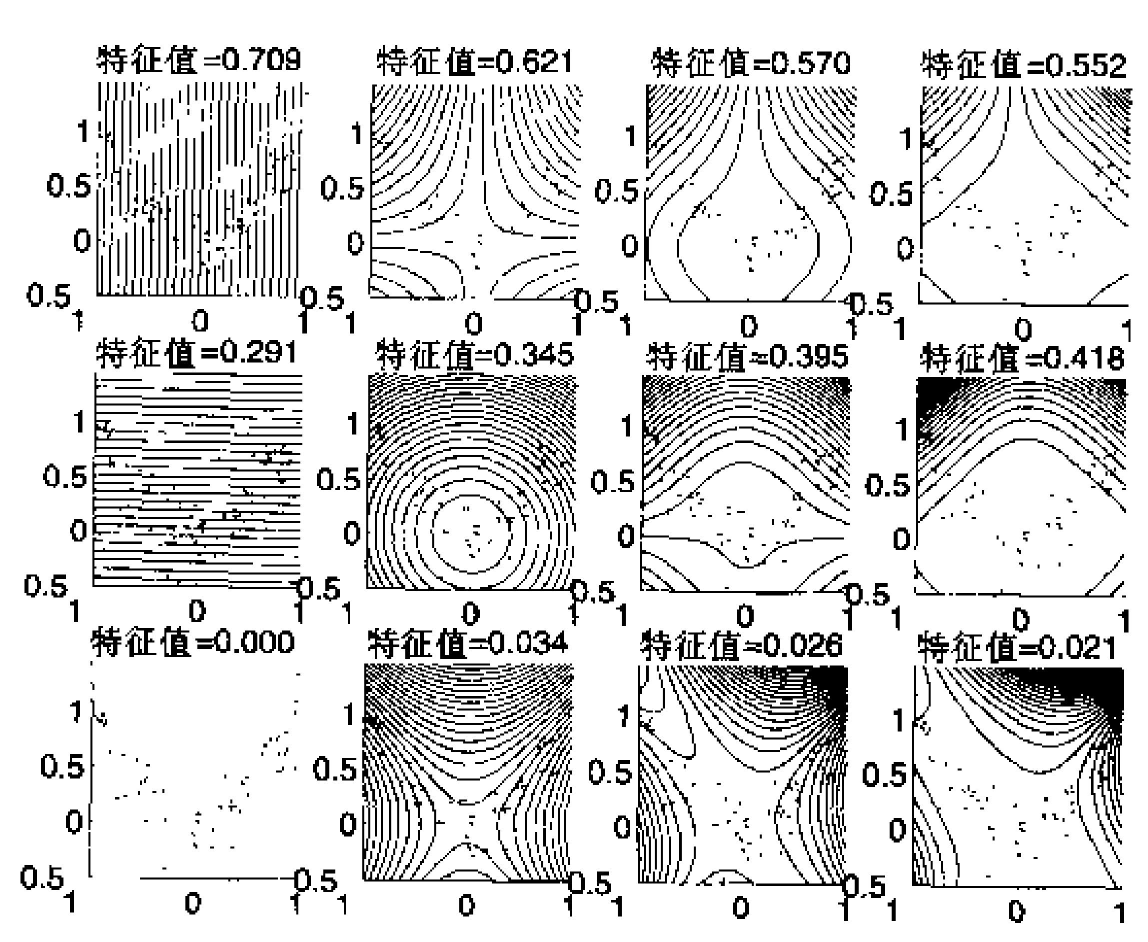


图 8-14 说明核 PCA 的二维示例。从左到右，核多项式的次数 $d=1,2,3,4$。从上到下，显示特征空间中的前面三个特征向量。第一列对应普通的 PCA，后三列对应多项式次数 $d=2,3,4$ 的核 PCA(复制经 Dr. Klaus-Robert Müller 允许。)

436

根据图 8-14 显示的结果可得到如下结论：

- 如所期望的，线性 PCA 不能对非线性输入数据提供足够的描述。
- 在所有情况下，第一个主分量沿着构成输入数据的抛物线单调变化。
- 在核 PCA 中，对不同的多项式次数 d，第 2 和第 3 个主分量展示一定的相似性。
- 在多项式次数 $d=2$ 情况下，核 PCA 的第 3 个主分量显现出找到加性高斯噪声 v 的方差。消除这个主分量的影响，在效果上实际是执行某种形式的噪声消除。

■

8.11 小结和讨论

在这一章中，我们提供处理主分量分析理论和用神经网络对其实现的材料。现在我们回顾这些材料并反问：主分量分析有多大用途？这个问题的答案当然依赖于有兴趣的应用。

如果主要目标是保存尽可能多的输入向量中的信息，并得到较好的*数据压缩*，则主分量分析提供一个有用的自组织学习过程。这里从 8.3 节的材料，我们注意到利用基于输入数据的“前 l 个主分量”的子空间分解方法所提供的线性映射，它在允许初始输入信号的重建按照

均方误差的意义下是最优化的。此外，基于前 l 个主分量的子空间表示比其他任意子空间表示更好，因为输入数据的主分量按特征值或方差递减顺序排列。因此，通过对输入数据的第一个主分量进行最精确的编码，对剩下的 $l-1$ 个主分量编码精度逐步降低，我们可以在数据压缩中最优化主分量分析的使用。

相关的问题是数据集的表示由几个聚类构成。因为聚类单独地可见，它们之间的间隔比聚类的内部散布要大。如果碰巧数据集中的聚类较少，用主分量分析建立的主要主轴将使聚类的投影具有好的分离，因此提供用于*特征提取*的有效基础。 437

在这后面的讨论中我们提及主分量分析器的有效应用——作为监督神经网络(例如反向传播训练的多层感知器)的*预处理器*。这里的动机是通过对输入数据去相关来加速学习过程的收敛。一个诸如反向传播算法的监督学习过程依赖于最速下降。因为多层感知器的突触权值对误差信号相互作用的效果，即使使用诸如对单个权值使用加入动量项和自适应学习率之类的简单局部加速过程，这种形式的学习过程收敛仍然特别慢。然而，如果多层感知的输入由不相关的分量组成，从第 4 章给出的讨论中我们注意代价函数$\mathscr{E}(n)$关于网络自由参数的 Hessian 矩阵将比在其他情况下更接近于对角化。因为有这种适当形式的对角化，则独立地沿着每个权值轴适当地提高学习率，用简单的局部加速过程就会使收敛过程有相当大的加速(Bercker, 1991)。

由于这一章基于 Hebb 的算法是由源于神经生物学的思想所激发，因此以对生物感知系统中主分量分析的作用的评论作为结束是合适的。Linsker(1990a)怀疑主分量分析作为一个原则的"充分性"，该原则用于决定通过单个神经元分析输入"场景"(scene)的一个总体所产生的响应性质。特别地，关于从神经元的响应实现对输入信号的精确重建与主分量分析最优性的相关性值得怀疑。一般地，人脑所做工作很显然比通过感觉单元的接收信号然后再简单重现输入场景复杂得多。相反，一些潜在的"有意义的线索"或特征被抽出来使得对输入得到高层的解释。因此我们可能对这个讨论开始时提出的问题加深了疑问，并且会问：主分量分析过程对感知过程到底有什么用处？

对于在*分层聚类算法*中由 Oja(1982)和 Sanger(1989a)建立用于主分量分析的算法(即 8.4 节和 8.5 节讨论的基于 Hebb 规则的算法)，Ambros-Ingerson et al.(1990)指出了它们的重要意义。他们提出假设认为分层聚类可以表现为基于长期潜能(lone-term potentiation, LTP)的记忆的基本性质(至少部分性质)，这个性质能够被用作识别环境的线索，所谓长期潜能就像在皮层球状网络发现的一类突触修改和在人脑其他区域里类似设计的回路。自组织主分量分析对在大脑皮层中学习线索的分层聚类具有重要意义，这一点并不是因为它的最优重建性质，而是由于其挑选的聚类投影具有好的分离间隔这一内在性质。

主分量分析在感觉处理中的另一个有趣的作用表现为*阴影成像*(shape-from-shading)问题的一个方法中，这是由 Atick et al.(1996)提出的。此问题可陈述如下：脑怎么能够从投影到二维图像的阴影模式感觉三维形状？Atick 等人提供一个阴影成像问题的分层解，包含两个概念： 438

1. 通过进化或先验经验，脑已经发现这样的物体，根据它们的形状就能分类成较低维的物体类。这个概念实际建立在这样一个事实的基础上，即脑用来抽取三维解释的*线索*是被透彻了解的。

2. 按照第一个概念，从阴影模式中抽取形状归结为低维空间中的*参数估计*这个更简单

的问题。

例如，人类头型的整个结构必然相同，在某种意义上所有的人都有凸出的鼻子，下陷的眼窝，平坦的前额和脸颊区域。这个不变性表明对任意给定的面部，在柱面(极)坐标上表示为 $r(\theta,l)$，可以用两部分和来表示：

$$r(\theta,l) = r_0(\theta,l) + \rho(\theta,l)$$

其中 $r_0(\theta,l)$表示对某类特定人(如成年男性或成年女性)的平均头(mean-head)，$\rho(\theta,l)$表示捕获特定人特征的扰动，通常 $\rho(\theta,l)$与 $r_0(\theta,l)$相比很小。Atick 等用主分量分析表示 $\rho(\theta,l)$，因此波动由一组特征函数表示(即特征向量的二维对应物)。Atick et al.(1996)的结果表明对某个人用这个人给定的一个二维图像，利用两阶段分层方法具有恢复3维曲面的能力。

注释和参考文献

[1] 在多元分析中，主分量分析(PCA)或许是最早的和最有名的方法(Jollife, 1986; Preisendorfer, 1988)。最早由 Pearson(1901)引入，在生物学背景下他用它来重建线性回归分析的新形式。后来 Hotelling(1933)在做心理测验时将它发展。看来 Karhunen(1947)年在概率论框架下再次独立地讨论了它；随后被 Loéve(1963)推广。

[2] Ljung(1977)和 Kushner and Clark(1978)研究随机逼近算法的动态行为所采取的措施归结为研究对应差分方程的动力学的问题。然而这两种方法根本不同。Ljung 的方法是利用 Lyapunov 函数，而 Kushner 和 Clark 采用的方法涉及线性插值过程和利用 Arzelà-Ascoli 定理(Dunford and Schwartz, 1966)。Kushner 和 Clark 的方法接着在 Diamantaras and Kung (1996)中被用于研究对基于 Hebb 的最大特征滤波器的收敛性。其中得到的结论与用 Ljung 方法得到的相同。

[3] Földiak(1989)扩展用于主分量分析的神经网络结构，引入反 Hebb 规则的反馈连接。这个修改的动机源于 Barlow and Földiak(1989)关于视觉皮层的自适应和去相关的早期工作；他们提出如果神经元按照反 Hebb 规则相互作用，则神经元输出定义一个坐标系统，在这个坐标系统中，即使输入具有很强的相关性，输出也不具有相关性。

439

Rubner and Tavan(1989) 和 Rubner and Schulten(1990)也提出在输出神经元中使用侧向抑制。然而，不像 Földiak 提出的模型，Rubner 等人考虑的侧向网络是不对称的连接。相反，侧向网络是分层的，其中(比如说)神经元 i 抑制除了$1,2,\cdots,i-1$外的所有神经元，其中 $i=1,2,\cdots$。

Kung and Diamantaras (1990)研究的 APEX 模型与 Rubner 等人的模型具有相同的网络拓扑，但是 Kung and Diamantaras (1990)的 APEX 模型在调整前馈和侧向连接的权值时均使用 Oja 的单个神经元学习规则(在 8.4 节描述)。

[4] Chen and Liu(1992)给出 APEX 算法收敛性的严格证明，所有的神经元趋于同时收敛。

[5] 讨论主分量分析的几个神经模型和它们的比较，请参看 Diamantaras and Kung(1996)的书。

[6] 非线性 PCA 方法，除了核 PCA 外，可以被归入三类网络(Diamantaras and Kung, 1996)：
- *Hebb 网络*，用非线性神经元代替基于 Hebb 规则的 PCA 算法的线性神经元得到。
- *复制器网络或自动编码器*，建立在多层感知器基础上：复制器网络在第4章讨论。
- *主曲线*，基于捕获数据结构的曲线或曲面的迭代估计(Hastie and Stuetzle, 1989)。在

Ritter et al.(1992)和 Cherkassky and Mulier(1995)中，指出 Kohonen 的自组织映射可被看作发现主曲线离散逼近的计算过程；自组织映射在下一章讨论。

习题

基于 Hebb 的最大特征滤波器

8.1 对于例 8.2 中考虑的匹配滤波器，特征值 λ_1 和对应的特征向量为 $\mathbf{q}_1$ 定义为

$$\lambda_1 = 1 + \sigma^2, \qquad \mathbf{q}_1 = \mathbf{s}$$

证明这些参数满足基本的关系

$$\mathbf{R}\mathbf{q}_1 = \lambda_1 \mathbf{q}_1$$

其中 $\mathbf{R}$ 为输入向量 $\mathbf{X}$ 的相关矩阵。

8.2 考虑最大特征滤波器，其中权值 $\mathbf{w}(n)$按照式(8.46)演化。证明随着 n 趋向于无穷大，滤波器的输出方差趋向于 λ_{max}，其中 λ_{max} 为输入向量相关矩阵的最大特征值。

8.3 次分量分析(minor components analysis, MCA)与主分量分析是相反的。在 MCA 中，我们寻找投影方差最小的方向。这样得到的方向对应于输入向量 $\mathbf{X}(n)$的相关矩阵 $\mathbf{R}$ 的最小特征值的特征向量。

440

在本题中，我们探讨怎样修改 8.4 节的单个神经元发现 $\mathbf{R}$ 的次分量。特别地，我们可以对式(8.40)的学习规则改变符号，得到(Xu et al., 1992)

$$w_i(n+1) = w_i(n) - \eta y(n)[x_i(n) - y(n)w_i(n)]$$

证明如果相关矩阵 $\mathbf{R}$ 的最小特征值 λ_m 重数为 1，则

$$\lim_{n\to\infty} \mathbf{w}(n) = \eta \mathbf{q}_m$$

其中 $\mathbf{q}_m$ 是与 λ_m 对应的特征向量。

基于 Hebb 的主分量分析

8.4 构造一个信号流图表示向量值等式(8.87)和(8.88)。

8.5 在 8.4 节描述的用于收敛性分析的常微分方程方法不能直接用于广义 Hebb 学习算法(GHA)。然而，通过将式(8.91)的突触权值矩阵 $\mathbf{W}(n)$用 $\mathbf{W}(n)$的列向量的组合来表示，则我们可以用通常的方式解释更新函数 $h(\cdot,\cdot)$，然后继续应用渐进稳定性定理。因此，根据此处已有的说明，证明 GHA 算法的收敛性定理。

8.6 在这个习题中，我们可以探讨利用广义 Hebb 算法来研究随机输入向量产生的二维接收域(Sanger, 1990)。随机输入包含独立于高斯噪声具有零均值和单位方差的二维域，它与高斯屏蔽(滤波器)作卷积，然后乘以一个高斯窗。高斯屏蔽有两个像素的标准偏差，高斯窗有 8 个像素的标准偏差。在位置(r,s)的结果随机输入 $x(r,s)$因而可以写成

$$x(r,s) = m(r,s)[g(r,s) * w(r,s)]$$

其中 $w(r,s)$是独立和同分布的高斯噪声的域，$g(r,s)$是高斯屏蔽，$m(r,s)$是窗函数。$g(r,s)$和 $w(r,s)$的循环卷积由

$$g(r,s) * w(r,s) = \sum_{p=0}^{N-1}\sum_{q=0}^{N-1} g(p,q)w(r-p, s-q)$$

定义，其中 $g(r,s)$和 $w(r,s)$均假设为周期的。

用随机输入 $x(r,s)$的 2000 个样本训练基于 GHA 算法的单层前馈网络。网络有 4096 个

输入，排列成64×64像素格网，具有16个输出。训练网络的结果突触权值用64×64阵列的数表示。执行上述计算并显示突触权值作为二维屏蔽的16个阵列。评价你的结果。

441

8.7 式(8.113)定义计算前馈权值向量 $\mathbf{a}_j(n)$ 的修正公式(8.106)的变换形式。变换基于由式(8.109)给出的网络的 m 主模式关于突触权值向量 $\mathbf{w}_j(n)$ 的定义。导出式(8.113)。

8.8 考虑式(8.116)的系统矩阵，它由图8-12的信号流图表示，对应于 $1 \leq k \leq j-1$。

(a)写出这个2×2矩阵的特征方程的公式。

(b)证明矩阵有一个二重特征值。

(c)证明结论：网络的所有主模式有相同的特征值。

8.9 GHA仅用前馈连接，而APEX算法使用前馈连接和侧向连接。尽管存在这些差别，在理论上APEX和GHA的长期收敛行为是相同的。证明这个结论的合理性。

核主分量分析

8.10 令 $\bar{K}_{ij}$ 表示核矩阵 $\mathbf{K}$ 的第 ij 个元素 K_{ij} 中心化后所对应的部分。证明(Schölkopf, 1997)

$$\bar{K}_{ij} = K_{ij} - \frac{1}{N}\sum_{m=1}^{N}\boldsymbol{\varphi}^T(\mathbf{x}_m)\boldsymbol{\varphi}(\mathbf{x}_j) - \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_n) + \frac{1}{N^2}\sum_{m=1}^{N}\sum_{n=1}^{N}\boldsymbol{\varphi}^T(\mathbf{x}_m)\boldsymbol{\varphi}(\mathbf{x}_n)$$

442

建议用紧凑的矩阵形式表示这个关系。

8.11 证明核矩阵 $\mathbf{K}$ 的特征向量 $\boldsymbol{\alpha}$ 的归一化与满足式(8.153)的条件等价。

8.12 小结核主分量分析的性质。

第9章　自组织映射

9.1　简介

在这一章我们通过考虑一种称为自组织映射的特殊人工神经网络继续研究自组织系统。这类网络基于竞争学习(competitive learning)；网络的输出神经元之间互相竞争以求被激活或点火，结果在每一时刻只有一个输出神经元，或者每组只有一个输出神经元被激活或点火。赢得竞争的一个输出神经元被称作*胜者全得*(winner-takes-all)*神经元*或简称*获胜*(winning)*神经元*。在输出神经元中导出胜者全得的竞争方法是在它们之间使用侧抑制连接(即负反馈路径)；这个思想是由 Rosenblatt(1958)最先提出的。

在*自组织映射*里，神经元被放置在*网格节点*上，这个网格通常是一维或是两维的。更高维映射也可以，但是不常见。在竞争学习过程中，神经元变化依不同输入模式(刺激)或者输入模式的类别而*选择性地调整*。这样调整后神经元(即获胜神经元)的位置彼此之间成为有序的，使得对于不同的输入特征，在网格上建立起有意义的坐标系(Kohonen,1990a)。因此自组织映射由输入模式的*拓扑映射*(topographic map)结构所表征，其中*网格神经元的空间位置表示输入模式包含的内在统计特征*，"自组织映射"因此得名。

作为一个神经模型，自组织映射在两个自适应层次之间提供一个桥梁：

- 在单个神经元的微观层次形成自适应规则。
- 在神经元层次的微观层上形成特征选择在实验上更好的和具体可实现的模式。

因为自组织映射本质上是非线性的，因此它被视为主分量分析的非线性推广(Ritter,1995)。

发展自组织映射作为神经模型是由人脑的一个突出特征所激发：人脑在许多地方以这样一种方式组织起来，使得不同的感觉输入由*拓扑有序的计算映射*(topologically ordered computational map)来表示。特别，感觉输入如触觉(Kaas et al.,1983)、视觉(Hubel and Wiesel,1962,1977)和听觉(Suga,1985)用拓扑有序的方式映射到人脑皮层的不同区域。这样在神经系统的信息处理基本结构中，计算映射组成一个基本构件。一个计算映射由神经元阵列定义，这些神经元表示略微不同调制的处理器和滤波器，它们并行处理携带信息的传感信号。所以，神经元将输入信号转变为*空间位置编码的概率分布*，分布通过映射中最大相关激活的位置表示参数的计算值(Knudsen et al.,1987)。用这种方式导出的信息属于这样一种形式，它可以用于使用相对简单的连接模式的高阶处理器。

本章的组织

这一章所讨论的关于计算映射的资料是按下面方式组织的。在 9.2 节，我们描述两个特征映射模型，它们用自己特有的方式解释或抓住人脑中计算映射的本质特征。两个模型使用的输入形式彼此不同。

本章其余各节详细地讨论这些模型中的一个，通常称为"自组织映射"，由 Kohonen(1982)提出。在 9.3 节里我们使用神经生物学的考虑方法建立 Kohonen 模型的一个数学公式。

该模型的小结在9.4节给出。模型的重要特性在9.5节描述，随后在9.6节讨论它的计算机仿真。特征映射的性能最终可能通过一个称为学习向量量化的监督技术进行微调；这个技术在9.7节讨论。9.8节描述一个关于自适应模式分类的计算机实验，它结合应用自组织映射和学习向量量化。在9.9节描述基于自组织映射的分层向量量化，它用于数据压缩。9.10节描述另一个自组织映射的应用，用于建立上下文映射，它从文本中音素类别的无监督分类、遥感和数据探索中找到应用。本章在9.12节给出一些最终评价作为结束。

9.2 两个基本的特征映射模型

任何人只要检查人脑就会禁不住对人脑被大脑皮质所占据的范围留下深深印象。人脑几乎完全被大脑皮质所包围，它遮蔽了其他部分。由于惊人的复杂性，大脑皮质也许超过了宇宙中任何已知的结构(Hubel and Wiesel,1977)。同样给我们深刻印象的是将不同的感觉输入(运动、身体的体觉、视觉、听觉等)以一种有序的方式映射到相应的大脑皮质区域的方法；为了说明这一点，看图2-4的大脑皮质的细胞结构图。计算映射的使用提供下面的特性(Knudsen et al.,1987)：

- 在表示的每一阶段，每一个新来的信息片段保持在它合适的位置中。
- 处理高度相关的信息片段的神经元被紧密地联系到一起，通过短的突触连接使得它们能够交互。

我们的兴趣在于建立人工拓扑映射，它以神经生物学激励的方式通过自组织来学习。在这段文字中，从人脑的计算映射的非常简短的讨论所体现的重要一点是拓扑映射构成原则，它可以陈述如下(Kohonen,1990a)：

在拓扑映射中输出神经元的空间位置对应于特殊的定义域或从输入空间抽取数据的特征。

这个原则提供了这里描述的两个基本不同的特征映射模型[1]的神经学生物基础。

图9-1展现两个模型的布局。在两种情况下输出神经元被安排在二维的网格中。这种拓扑确保每个神经元都有一组邻域。模型间的区别在于输入模式的指定方式。

图9-1a的模型由Willshaw and von der Malsburg(1976)在生物学基础上首先提出的，用以解释(在高级脊椎动物中)从视网膜到视觉皮质的视觉映射的问题。具体地，有两个不同的二维网格神经元连接在

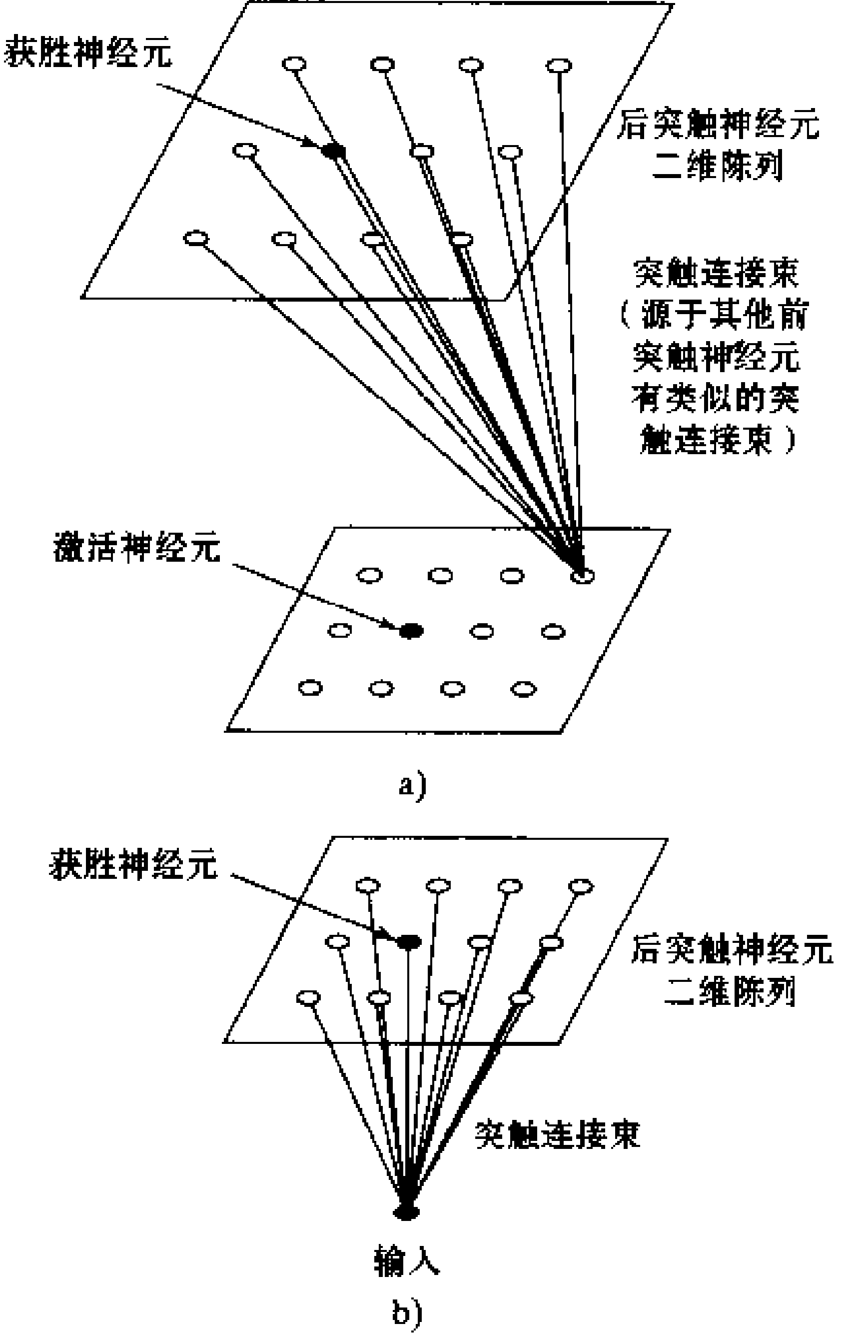


图9-1 两个自组织特征映射
a) Willshaw-von der Malsburg模型 b) Kohonen模型

一起，一个投射到另一个。一个网格代表前突触(输入)神经元，另一个网格代表后突触(输出)神经元。后突触网格使用短程兴奋机制(short-range excitatory mechanism)和长程抑制机制(long-range inhibitory mechanism)。这两种机制本质上都是局部的且对自组织特别重要。这两个网格由Hebb型的可调突触相互连接。因此严格地说，后突触神经元并不是胜者全得；相反使用阈值确保在任一时刻仅有一些后突触神经元点火。更进一步，为了防止可能导致网络不稳定性的突触权值的稳定建立，每个后突触神经元的总权值有一个上界[2]。因此对每个神经元一些突触权值上升伴随着另外的神经元下降。Willshaw-von der Malsburg模型的基本思想是对前突触神经元的几何邻近编码为它们电位活动的相关形式，并且在后突触网格中利用这些相关使得相邻的前突触神经元连接到相邻的后突触神经元。从而由自组织产生拓扑有序的映射。但需注意Willshaw-von der Malsburg模型限制为输入和输出维数相同的映射。

图9-1b的第二个模型，由Kohonen(1982)引入，并不在说明神经生物学的细节。模型抓住人脑中计算映射的本质特征而且保留计算的易行性[3]。Kohonen模型看起来比Willshaw-von der Malsburg模型更为一般，前者能进行数据压缩(即输入维数的缩减)。

现实中，Kohonen模型属于向量-编码(vector-coding)算法的类型。模型提供一个拓扑映射，它最优地设置固定数目的向量(即编码字)到高维输入空间，因此有利于数据压缩。Kohonen模型因此可由两种方式导出。我们可以用由神经生物学考虑所激发的自组织的基本思想导出模型，这是传统的方法(Kohonen,1982,1990a,1997a)。另外，可以用向量量化的方法，使用包含编码器和解码器的模型，这由通信理论的考虑所激发。在这一章我们考虑这两种方法。

在文献中Kohonen模型比Willshaw-von der Malsburg模型受到更多的注意。它拥有在本章后面讨论的一些性质，这使得它对人脑中的皮质映射的理解和建模有特殊的兴趣。本章剩余部分介绍自组织映射的导出、它基本性质和细节。

9.3 自组织映射

自组织映射(self-organizing map,SOM)的主要目的是将任意维数的输入信号模式转变为一维或二维的离散映射，并且以拓扑有序的方式自适应实现这个变换。图9-2给出常用作离散映射的二维神经元网格的简要图表。网格中每个神经元和输入层的源节点全连接。这个网络代表具有神经元按行和列构成的单一计算层的前馈结构。一维网格是图9-2描绘的构形的一个特例：在这种特殊情形计算层仅由单一的行或列神经元构成。 446

呈现给网络的每个输入模式，通常包含面对平静背景的一个局部化活动区域或“点”。这个点的位置和性质通常随输入模式的实现不同而不同。因此输入网络中所有神经元应经历输入模式的足够次数的不同实现，确保有机会完成恰当的自组织过程。

负责形成自组织映射的算法，第一步进行网络突触权值的初始化。这个工作可以从随机数产生器中挑选较小的值赋予它们；这样做，在特征映射上没有加载任何先验的序。一旦网络被恰当初始化，在自组织映射的形成中有三个主要过程，小结如下：

1. 竞争。对每个输入模式，网络中的神经元计算它们各自的判别函数的值。这个判别函数对神经元之间的竞争提供基础。具有判别函数最大值的特定神经元成为竞争的胜利者。

2. 合作。获胜神经元决定兴奋神经元的拓扑邻域的空间位置，从而提供这样的相邻神经元合作的基础。 447

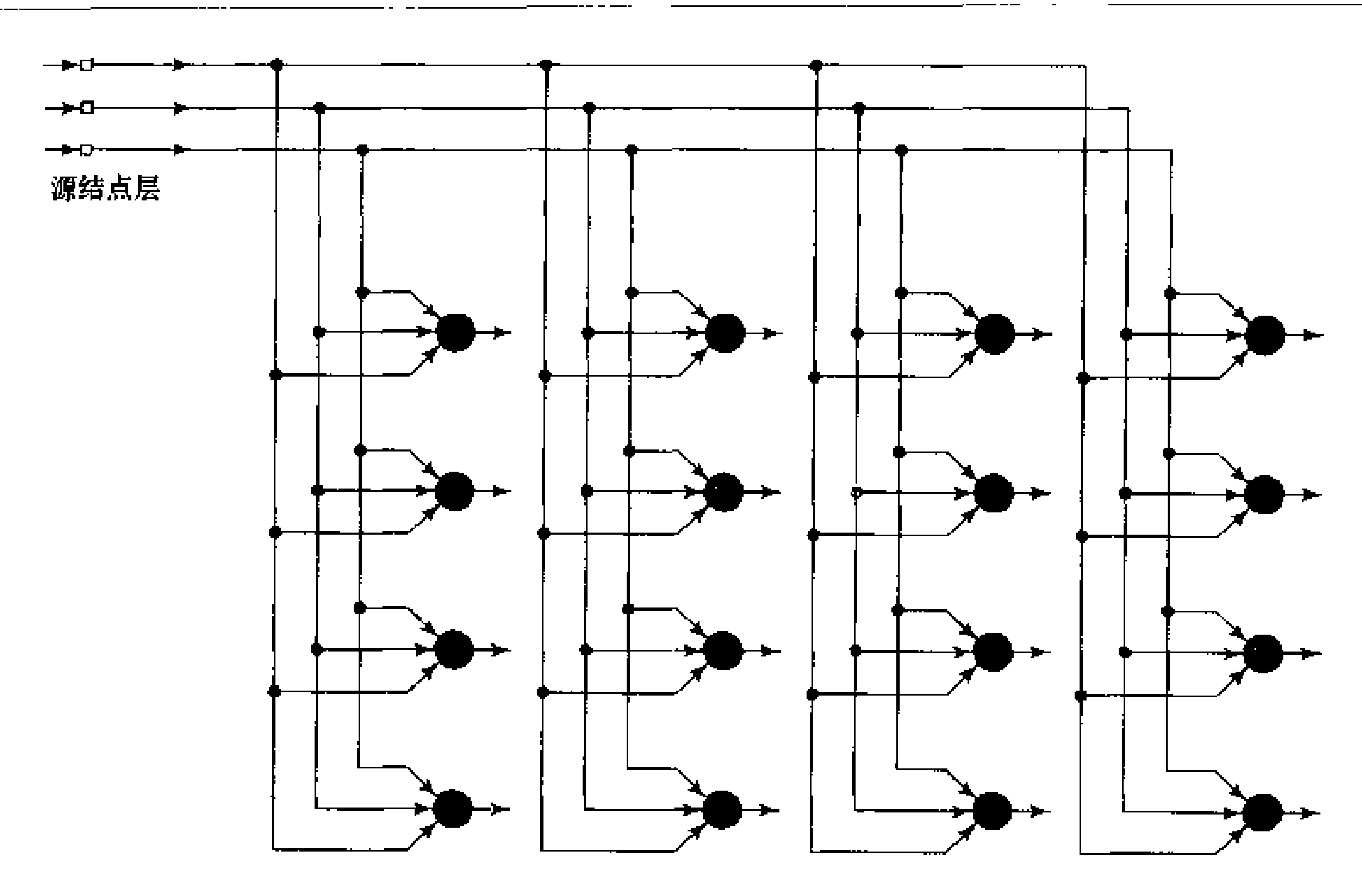


图 9-2 神经元的二维网格

3. 突触调节。最后的这个机制使兴奋神经元通过对它们突触权值的适当调节以增加它们关于该输入模式的判别函数值。所做的调节使获胜神经元对以后相似输入模式的响应增强了。

竞争和合作的过程符合第 8 章描述的四个自组织原则中的两个。对于自增强原则，它来源于自适应过程的 Hebb 学习的修正形式。如第 8 章的解释，输入数据中的冗余(虽然在描述 SOM 算法时没有明显提及)对学习是需要的，因为它提供知识。现在给出竞争、合作和突触调节过程的详细描述。

竞争过程

令 m 表示输入(数据)空间的维数。从输入空间中随机选择输入模式(向量)记为

$$\mathbf{x} = [x_1, x_2, \cdots, x_m]^T \tag{9.1}$$

网络中每个神经元的突触权值向量和输入空间的维数相同。神经元 j 的突触权值向量记为

$$\mathbf{w}_j = [w_{j1}, w_{j2}, \cdots, w_{jm}]^T, \qquad j = 1,2,\cdots,l \tag{9.2}$$

其中 l 是网络中神经元的总数。为了找到输入向量 $\mathbf{x}$ 与突触权值向量 $\mathbf{w}_j$ 的最好匹配，对 $j = 1,2,\cdots,l$ 比较内积 $\mathbf{w}_j^T\mathbf{x}$ 并选择最大者。这里假定所有的神经元有相同的阈值；阈值是偏置取负。这样，通过选择具有最大内积 $\mathbf{w}_j^T\mathbf{x}$ 的神经元，我们实际上决定了兴奋神经元的拓扑邻域中心的位置。

从第 1 章我们回想基于内积 $\mathbf{w}_j^T\mathbf{x}$ 最大化的最优匹配准则，在数学上等价于向量 $\mathbf{x}$ 和 $\mathbf{w}_j$ 的 Euclid 距离的最小化。如果用标号 $i(\mathbf{x})$ 标识最优匹配输入向量 $\mathbf{x}$ 的神经元，我们可以通过下列条件[4]决定 $i(\mathbf{x})$：

448

$$i(\mathbf{x}) = \arg\min_j \| \mathbf{x} - \mathbf{w}_j \|, \quad j = 1,2,\cdots,l \tag{9.3}$$

这概括了神经元中竞争过程的本质。根据式(9.3)，$i(\mathbf{x})$是注意的目标，因为我们要识别神经元 i。满足这个条件的特定神经元 i 被称为输入向量 $\mathbf{x}$ 的神经元或获胜神经元。式(9.3)导出这样的观察：

激活模式的连续输入空间通过网络中神经元之间的竞争过程映射到神经元的离散输出空间。

根据应用的不同，网络的响应可能是获胜神经元的标号(即它在网格中的位置)或者是在Euclid距离意义下距输入向量最近的突触权值向量。

合作过程

获胜神经元位于合作神经元的拓扑邻域的中心。关键问题是：我们怎样定义一个在神经生物学上正确的拓扑邻域？为了回答这个问题，记住对于一组兴奋神经元的侧向相互作用有神经生物学的证据。具体地，一个点火的神经元倾向于激活它紧接的邻域内的神经元而不是和它隔得远的神经元，这在直观上是满足的。这个观察引导我们对获胜神经元的拓扑邻域按侧向距离光滑地缩减[5](Lo et al., 1991, 1993; Ritter et al., 1992)。具体地，设 $h_{j,i}$表示以获胜神经元 i 为中心的拓扑邻域。设 $d_{i,j}$表示在获胜神经元 i 和兴奋神经元 j 的侧向距离。然后我们可以假定拓扑邻域 $h_{j,i}$是侧向距离 $d_{j,i}$的单峰函数使得它满足两个不同的要求：

- 拓扑邻域 $h_{j,i}$关于 $d_{i,j}=0$ 定义的最大点是对称的；换句话说，在距离 $d_{j,i}$为零的获胜神经元 i 处达到最大值。
- 拓扑邻域 $h_{j,i}$的幅度值随侧向距离 $d_{i,j}$的增加而单调递减，当 $d_{j,i}\to\infty$ 时趋于零；对收敛来说这是一个必要条件。

满足这些要求的一个 $h_{j,i}$的典型选择为高斯函数[6]

$$h_{j,i(\mathbf{x})}=\exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right) \tag{9.4}$$

它是平移不变的(即不依赖于获胜神经元的位置)。图 9-3 所示参数 σ 是拓扑邻域的“有效宽度”；它度量靠近获胜神经元的兴奋神经元在学习过程中参与的程度。就量化来说，式(9.4)所示的高斯拓扑邻域比矩形形式的拓扑邻域在生物上更合适。它的使用使 SOM 算法的收敛速度比矩形拓扑邻域更快(Lo et al., 1991, 1993; Erwin et al., 1992a)。

对于邻域函数神经元之间的合作，必然要求拓扑邻域函数 $h_{j,i}$依赖获胜神经元 i 和兴奋神经元 j 在输出空间的侧向距离 $d_{j,i}$而不是依赖于原始输入空间的某种距离度量。这正是在式(9.4)中我们所表达的意义。就一维网格来说，$d_{j,i}$是整数且等于 $|j-i|$。另一方面，在两维网格形的情况它定义为

$$d_{j,i}^2=\|\mathbf{r}_j-\mathbf{r}_i\|^2 \tag{9.5}$$

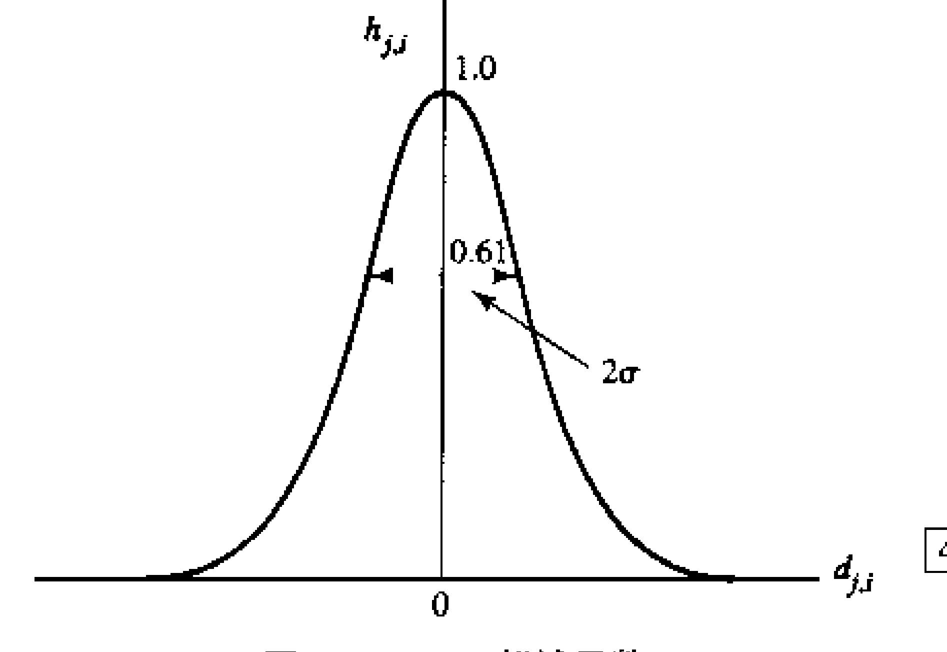


图 9-3 Gauss 邻域函数

449

其中离散向量 $\mathbf{r}_j$ 定义兴奋神经元 j 的位置，而 $\mathbf{r}_i$ 定义获胜神经元 i 的离散位置，两者都是在离散输出空间中度量的。

SOM 算法的另一个独有特征是拓扑邻域的大小随时间收缩。这个要求通过使拓扑邻域函数 $h_{j,i}$ 的宽度 σ 随时间而下降来满足。对于 σ 依赖于离散时间 n 的流行选择是由

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right), \qquad n = 0,1,2,\cdots, \tag{9.6}$$

描述的指数衰减(Ritter et al.,1992;Obermayer et al.,1991)，其中 σ_0 是 SOM 算法中 σ 的初值，τ_1 是时间常数。因此，拓扑邻域假定具有时变形式，表示如下

$$h_{j,i(\mathbf{x})}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right), \qquad n = 0,1,2,\cdots, \tag{9.7}$$

其中 $\sigma(n)$ 由式(9.6)定义。于是随着 n(即迭代次数)的增加宽度 $\sigma(n)$ 以指数下降，拓扑邻域以相应的方式缩减。这样我们将 $h_{j,i(\mathbf{x})}(n)$ 称作*邻域函数*。

[450] 另一种关于邻域函数 $h_{j,i(\mathbf{x})}(n)$ 在获胜神经元 $i(\mathbf{x})$ 周围变动的有用观点如下(Luttrell,1989a)。宽的 $h_{j,i(\mathbf{x})}(n)$ 的目标是使网格中大量兴奋神经元的权值更新方向相关。随着 $h_{j,i(\mathbf{x})}(n)$ 宽度减少，更新方向相关的神经元数量也在减少。当自组织映射的训练在计算机图形屏幕显示时，这个现象尤其明显。以相关形式在获胜神经元周围移动大量自由度是相当耗费计算机资源的，就像标准 SOM 算法一样。相反，使用*重正规化*(renormalized)SOM 的训练形式会更好，这样我们工作在较小数量的*正规化自由度*上。通过使用恒定宽度的邻域函数 $h_{j,i(\mathbf{x})}(n)$，但逐渐增加神经元的数量，这个操作很容易以离散形式完成。新的神经元被插到已有的神经元之间，而 SOM 算法的平滑性保证新的神经元以很好的方式参与突触自适应(Luttrell,1989a)。重正规化 SOM 算法的概述在习题 9.13 给出。

自适应过程

现在我们来讨论特征映射自组织形成过程的最后一个过程，即突触自适应过程。为了使网络成为自组织的，要求神经元 j 的突触权值向量 $\mathbf{w}_j$ 随输入向量 $\mathbf{x}$ 改变。问题是怎样作改变。在 Hebb 学习假设中，突触权值随着前突触和后突触的激活同时发生而增加。此方法非常适合联想学习。然而对于这里考虑的无监督学习，以 Hebb 假设的基本形式是不能令人满意的，原因如下：连接的改变仅发生在一个方向上，这样最终使所有的突触权值都趋于饱和。为了克服这个问题，我们通过包括一个*遗忘项* $g(y_j)\mathbf{w}_j$ 来改变 Hebb 假定，其中 $\mathbf{w}_j$ 是神经元 j 的突触权值向量，$g(y_j)$ 是响应 y_j 的正的标量函数。对 $g(y_j)$ 的惟一强制要求是它的 Taylor 级数展开的常数项为零，这样我们可写成

$$g(y_j) = 0 \qquad 对于\ y_j = 0 \tag{9.8}$$

这个要求的意义很快就会变得明显。给定这样一个函数，我们可以把网格中神经元 j 的权值向量改变表示成

$$\Delta\mathbf{w}_j = \eta y_j \mathbf{x} - g(y_j)\mathbf{w}_j \tag{9.9}$$

其中 η 是算法的*学习率参数*。右端第一项是 Hebb 项，第二项是遗忘项。为了满足式(9.8)，对 $g(y_j)$ 选择线性函数如下：

$$g(y_j) = \eta y_j \tag{9.10}$$

我们可以进一步简化式(9.9)，置

$$y_j = h_{j,i(\mathbf{x})} \tag{9.11}$$

用式(9.10)和(9.11)代入式(9.9)得到

$$\Delta\mathbf{w}_j = \eta h_{j,i(\mathbf{x})}(\mathbf{x} - \mathbf{w}_j) \tag{9.12}$$

最后使用离散时间形式，假定在时刻 n 神经元 j 的权值向量为 $\mathbf{w}_j(n)$，更新权值向量 $\mathbf{w}_j(n+1)$ 在时刻 $n+1$ 被定义为(Kohonen, 1982; Ritter et al., 1992; Kohonen, 1997a)：

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n) h_{j,i(\mathbf{x})}(n)(\mathbf{x} - \mathbf{w}_j(n)) \tag{9.13}$$

451

它被应用到网格中获胜神经元 i 的拓扑邻域中的所有神经元。式(9.13)具有将获胜神经元 i 的突触权值向量 $\mathbf{w}_i$ 向输入向量 $\mathbf{x}$ 移动的作用。随着训练数据的重复出现，由于邻域更新使得突触权值向量趋于服从输入向量的分布。因此算法导致在输入空间中特征映射的拓扑排序，这意味着网格中相邻神经元会有相似的突触权值向量。关于这一点在 9.5 节中，我们将进一步详述。

式(9.13)为计算特征映射突触权值所期望的公式。除了这个公式之外，我们还需要用于选择邻域函数 $h_{j,i(\mathbf{x})}(n)$ 的启发式规则(9.7)式和另一个用于选择学习率参数 $\eta(n)$ 的启发式规则。

学习率参数 $\eta(n)$ 应如式(9.13)所示的时变形式，这也是它用于随机逼近的要求。特别地，它应从初始值 η_0 开始，然后随时间 n 增加而逐渐下降。这个要求可以通过选择 $\eta(n)$ 指数衰减而满足，表示为

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right), \quad n = 0, 1, 2, \cdots, \tag{9.14}$$

其中，τ_2 是 SOM 算法的另一个时间常数。即使在式(9.6)和(9.14)中描述的邻域函数宽度和学习率参数分别以指数衰减的公式可能不是最优的，但它们对于以自组织方式构成特征映射是足够的。

自适应过程的两个阶段：排序和收敛

假定算法的参数是正确选择的，从完全无序的初始状态开始，SOM 算法怎样逐步导致一个从输入空间抽取的激活模式的有组织表示，这是令人惊奇的。我们可以把根据式(9.13)计算的网络权值的自适应分解为两个阶段：排序或自组织阶段及其后的收敛阶段。自适应过程的这两个阶段描述如下(Kohonen, 1982, 1997a)：

1. 自组织或排序阶段。在自适应过程的第一阶段形成权值向量的拓扑排序。这个排序阶段可能需要 SOM 算法的 1000 次迭代，也许会更多。要仔细考虑学习率参数和邻域函数的选择：

- 学习率参数 $\eta(n)$ 初始值应接近 0.1；然后逐渐减少，但应保持在 0.01 以上。这些要求的值可以在公式(9.14)中选择 $\eta_0 = 0.1$，$\tau_2 = 1000$ 得到满足。

452

- 邻域函数 $h_{j,i}(n)$ 的初始化应包括以获胜神经元 i 为中心的几乎所有神经元，然后随时间慢慢收缩。尤其，排序阶段可能需要 SOM 算法的 1000 次迭代或更多，仅仅对一些神经元或获胜神经元本身允许 $h_{j,i}(n)$ 减少到很小的值。假定对离散映射使用神

经元二维网格，则我们可以设定邻域函数的初始值 σ_0 等于网格的半径。相应地我们设定式(9.6)的时间常数 $\tau_1 = 1000/\log\sigma_0$。

2. *收敛阶段*：自适应过程的第二阶段需要微调特征映射从而提供输入空间的准确统计量。作为一般性规则，组成收敛阶段的迭代次数至少是网络中神经元数目的500倍。这样收敛阶段可能进行几千次以至上万次的迭代：

- 对于好的统计精度，在收敛阶段学习参数 $\eta(n)$ 应该保持在较小的值上，为0.01数量级。无论如何，不允许它下降到零；否则，网络会陷入到亚稳定状态。*亚稳定状态*(metastable state)属于有拓扑缺陷的特征映射结构。式(9.14)的指数衰减保证不可能进入亚稳定状态。
- 邻域函数 $h_{j,i(\mathbf{x})}$ 应该仅包括获胜神经元的最近邻域，最终减到一个或零个邻域神经元。

9.4 SOM算法小结

Kohonen的SOM算法的本质是它用一个简单的几何计算代替类Hebb规则的复杂性质和侧向相互作用。算法的主要构成/参数有：

- 根据一定概率分布产生激活模式的连续输入空间。
- 以神经元的网格形式表示的网络拓扑，它定义一个离散输出空间。
- 在获胜神经元 $i(\mathbf{x})$ 周围定义随时间变化的邻域函数 $h_{j,i(\mathbf{x})}(n)$。
- 学习率参数 $\eta(n)$ 的初始值是 η_0，然后随着时间 n 递减，但永不为零。

453

对于邻域函数和学习率参数，在排序阶段(即开始的大约1000次迭代)我们分别使用式(9.7)和(9.14)。为了好的统计精度，在收敛阶段 $\eta(n)$ 在相当长的时间内应该保持一个较小值(0.01或更小)，一般为几千次迭代。对于邻域函数，在收敛阶段之初，它应仅包含获胜神经元的最近的领域，并且最终缩减到一个或零个邻域神经元。

在初始化后算法的应用中涉及三个基本步骤：取样，相似性匹配，更新。重复这三个步骤直到完成特征映射的形成。算法小结如下：

1. *初始化*。对初始权值向量 $\mathbf{w}_j(0)$ 选择随机值。这里惟一的限制是对 $j = 1, 2, \cdots, l$，$\mathbf{w}_j(0)$ 互不相同，其中 l 是网格中神经元的数目。可能希望保持较小的权值。

另一种算法初始化方法是从输入向量 $\{\mathbf{x}_i\}_{i=1}^N$ 的可用集里随机选择权值向量 $\{\mathbf{w}_j(0)\}_{j=1}^l$。

2. *取样*。以一定概率从输入空间取样本 $\mathbf{x}$；向量 $\mathbf{x}$ 表示应用于网格的激活模式。向量 $\mathbf{x}$ 的维数等于 m。

3. *相似性匹配*。在时间步 n 使用最小Euclid距离准则寻找最匹配(获胜)的神经元 $i(\mathbf{x})$：

$$i(\mathbf{x}) = \arg\min_j \|\mathbf{x}(n) - \mathbf{w}_j\|, j = 1, 2, \cdots, l$$

4. *更新*。通过用更新公式

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n) h_{j,i(\mathbf{x})}(n)(\mathbf{x}(n) - \mathbf{w}_j(n))$$

调整所有神经元的权值向量，其中 $\eta(n)$ 是学习率参数，$h_{j,i(\mathbf{x})}(n)$ 是获胜神经元 $i(\mathbf{x})$ 周围的邻域函数；为了获得最好的结果，$\eta(n)$ 和 $h_{j,i(\mathbf{x})}(n)$ 在学习过程中是动态变化的。

5. *继续*。继续步骤2直到在特征映射里观察不到明显的变化为止。

9.5 特征映射的性质

一旦 SOM 算法收敛，由算法计算的特征映射显示输入空间的重要统计特性。

开始令$\mathscr{X}$表示空间的连续输入(数据)空间，它的拓扑由向量 $\mathbf{x}\in\mathscr{X}$的度量关系定义。令$\mathscr{A}$表示空间的离散输出空间，其拓扑由安排一组神经元作为网格的计算节点来赋予。令 Φ 表示称为特征映射的非线性变换，它映射输入空间$\mathscr{X}$到输出空间$\mathscr{A}$，表示为

$$\Phi:\mathscr{X}\rightarrow\mathscr{A} \tag{9.15}$$

454

式(9.15)可看成式(9.3)的抽象，式(9.3)定义为响应输入向量 $\mathbf{x}$ 而产生的获胜神经元 $i(\mathbf{x})$的位置。例如，在神经生物学中输入空间$\mathscr{X}$可以表示密布于整个体表面的体感觉接受器的坐标集。相应地，输出空间$\mathscr{A}$表示位于限制体感觉接受器的人脑皮层中的神经元集。

给定输入向量 $\mathbf{x}$，SOM 算法首先根据特征映射 Φ 确定在输出空间$\mathscr{A}$中的最佳匹配或获胜神经元。神经元 $i(\mathbf{x})$的突触权值向量 $\mathbf{w}_i$ 可以视为神经元指向输入空间的指针；即向量 $\mathbf{w}_i$ 的突触元素可以视为神经元 i 投影到输入空间的图像坐标。这两个操作在图 9-4 中描绘。特征映射 Φ 有某些重要性质：

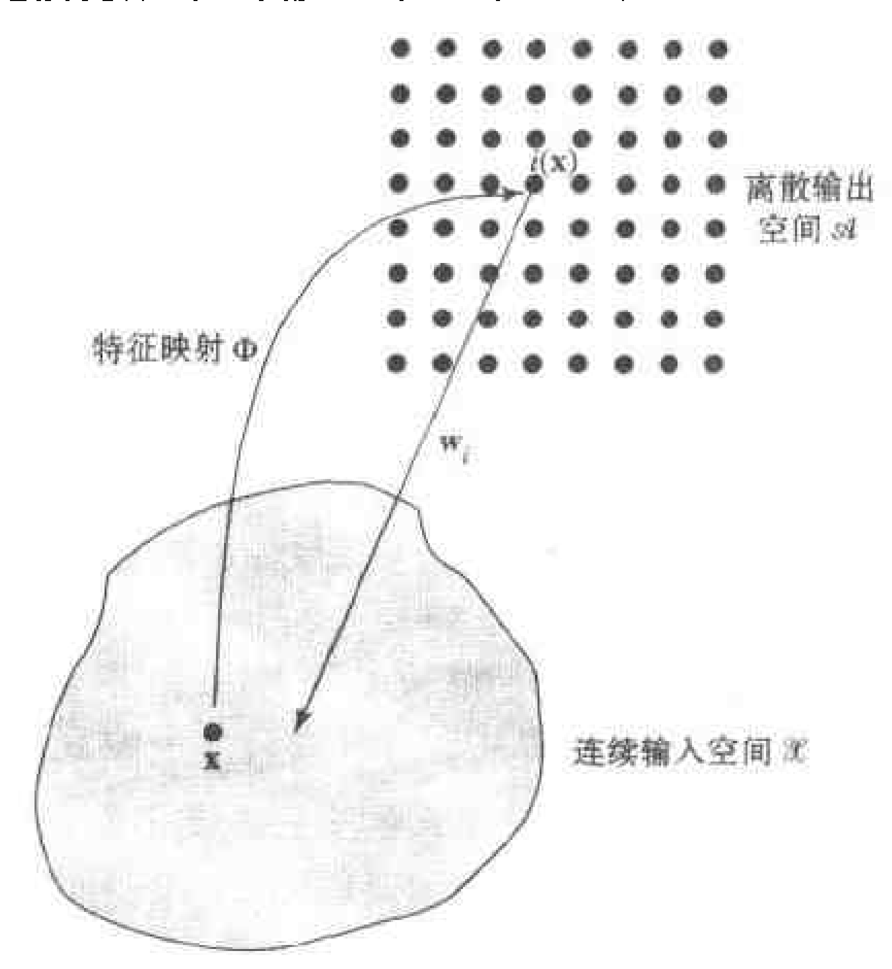


图 9-4 特征映射 Φ 和获胜神经元 i 权值向量 $\mathbf{w}_i$ 的关系图

性质 1 输入空间的近似 由输出空间$\mathscr{A}$的突触权值向量$\{\mathbf{w}_j\}$的集合表示的特征映射 Φ 对输入空间$\mathscr{X}$提供一个好的近似。

SOM 算法的基本目标是通过寻找原型 $\mathbf{w}_j\in\mathscr{A}$的一个较小的集合存储输入向量 $\mathbf{x}\in\mathscr{X}$的一个大集合，从而对原始输入空间$\mathscr{X}$提供一个好的近似。刚才描述的思想的理论基础植根于向量量化理论(vector quantization theory)，它的动机是维数的削减或者是数据的压缩(Gersho and Gray, 1992)。因此给出这个理论的简要讨论是适宜的。

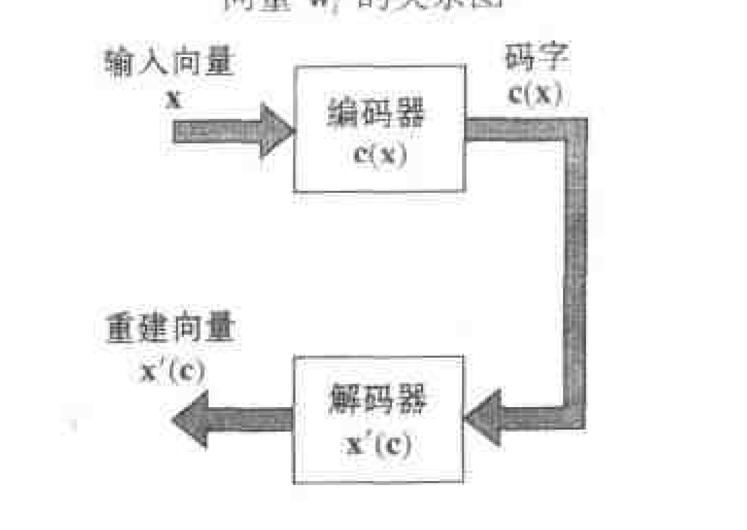


图 9-5 编码器-解码器模型

455

考虑图 9-5，其中 $\mathbf{c}(\mathbf{x})$作为输入向量 $\mathbf{x}$ 的编码器而 $\mathbf{x}'(\mathbf{c})$作为 $\mathbf{c}(\mathbf{x})$的解码器。向量 $\mathbf{x}$ 从满足固有概率密度函数 $f_{\mathbf{X}}(\mathbf{x})$的训练样本(即输入空间$\mathscr{X}$)中随机选择。通过变化函数 $\mathbf{c}(\mathbf{x})$和 $\mathbf{x}'(\mathbf{c})$决定最优编码-解码方案使得极小化由

$$D=\frac{1}{2}\int_{-\infty}^{\infty}d\mathbf{x}f_{\mathbf{X}}(\mathbf{x})d(\mathbf{x},\mathbf{x}') \tag{9.16}$$

定义的期望失真，其中引入因子1/2是为了表达方便，$d(\mathbf{x},\mathbf{x}')$是失真(distortion)度量。积分在假定维数为m的整个输入空间$\mathscr{X}$上进行。失真度量$d(\mathbf{x},\mathbf{x}')$的一个常用选择是输入向量$\mathbf{x}$和重建向量$\mathbf{x}'$之间的Euclid距离的平方；即

$$d(\mathbf{x},\mathbf{x}') = \|\mathbf{x}-\mathbf{x}'\|^2 = (\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}') \tag{9.17}$$

这样我们可把式(9.16)重写为

$$D = \frac{1}{2}\int_{-\infty}^{\infty} d\mathbf{x} f_{\mathbf{X}}(\mathbf{x}) \|\mathbf{x}-\mathbf{x}'\|^2 \tag{9.18}$$

期望失真D最小化的必要条件在广义Lloyd算法[7]中(Gersho and Gray,1992)。条件是两方面的：

条件1. 给定输入向量$\mathbf{x}$，选择码字$\mathbf{c}=\mathbf{c}(\mathbf{x})$使其最小化平方误差失真$\|\mathbf{x}-\mathbf{x}'(\mathbf{c})\|^2$。

条件2. 给定码字$\mathbf{c}$，计算重构向量$\mathbf{x}'=\mathbf{x}'(\mathbf{c})$作为满足条件1的输入向量$\mathbf{x}$的中心。

条件1称为最近邻编码规则。条件1和2意味着平均失真D关于编码器$\mathbf{c}(\mathbf{x})$和解码器$\mathbf{x}'(\mathbf{c})$各自的变化是稳定的。为了实现向量量化，广义Lloyd算法以集中方式运行。基本上，算法包含交替按照条件1优化编码器$\mathbf{c}(\mathbf{x})$和按照条件2优化解码器$\mathbf{x}'(\mathbf{c})$，直到期望失真D达到一个最小。为了克服局部最小问题，可能需要以不同初值运行广义Lloyd算法若干次。

456

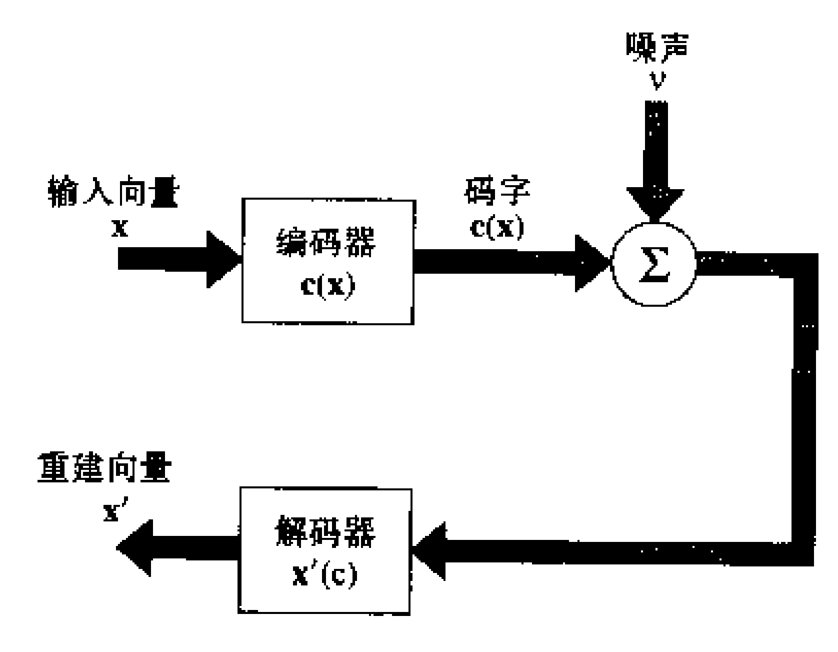


图9-6 噪声编码器－解码器模型

广义Lloyd算法和SOM算法紧密相关，如Luttrell(1989b)所示。可以通过考虑图9-6所示的系统描述这种关系的形式，其中在编码器$\mathbf{c}(\mathbf{x})$之后我们引入了独立于数据的噪声过程$\boldsymbol{\nu}$。噪声$\boldsymbol{\nu}$附加在编码器和解码器之间的虚构的“通信信道”上，它的目的是说明输出码字$\mathbf{c}(\mathbf{x})$可能失真的可能性。在图9-6所示模型的基础上，可以考虑期望失真的一种修正形式

$$D_1 = \frac{1}{2}\int_{-\infty}^{\infty} d\mathbf{x} f_{\mathbf{X}}(\mathbf{x}) \int_{-\infty}^{\infty} d\boldsymbol{\nu}\,\pi(\boldsymbol{\nu}) \|\mathbf{x}-\mathbf{x}'(\mathbf{c}(\mathbf{x})+\boldsymbol{\nu})\|^2 \tag{9.19}$$

其中$\pi(\boldsymbol{\nu})$为加性噪声$\boldsymbol{\nu}$的概率密度函数(pdf)，第二个积分是对这个噪声的所有可能实现。

根据广义Lloyd算法描述的策略，对图9-6所示的模型可考虑两个不同的优化，一个属于编码器而另一个属于解码器。为了找到给定$\mathbf{x}$的最优编码器，我们需要期望失真度量D_1对编码向量$\mathbf{c}$的偏导数。利用(9.19)，可得

$$\frac{\partial D_1}{\partial \mathbf{c}} = \frac{1}{2} f_{\mathbf{X}}(\mathbf{x}) \int_{-\infty}^{\infty} d\boldsymbol{\nu}\,\pi(\boldsymbol{\nu}) \frac{\partial}{\partial \mathbf{c}} \|\mathbf{x}-\mathbf{x}'(\mathbf{c})\|^2 \Big|_{\mathbf{c}=\mathbf{c}(\mathbf{x})+\boldsymbol{\nu}} \tag{9.20}$$

为了找到给定$\mathbf{c}$的最优解码器，我们需要期望失真度量D_1对解码向量$\mathbf{x}'(\mathbf{c})$的偏导数。利用式(9.19)，可得

$$\frac{\partial D_1}{\partial \mathbf{x}'(\mathbf{c})} = -\int_{-\infty}^{\infty} d\mathbf{x} f_{\mathbf{X}}(\mathbf{x}) \pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))(\mathbf{x}-\mathbf{x}'(\mathbf{c})) \tag{9.21}$$

因此，根据式(9.20)和(9.21)，以前陈述的广义Lloyd算法的条件1和条件2必须修改如下

(Luttrell, 1989b):

条件 I. 给定输入向量 $\mathbf{x}$, 选择码字 $\mathbf{c}=\mathbf{c}(\mathbf{x})$ 使其最小化失真度量

$$D_2=\int_{-\infty}^{\infty}d\nu\pi(\nu)\|\mathbf{x}-\mathbf{x}'(\mathbf{c}(\mathbf{x})+\nu)\|^2 \tag{9.22}$$

457

条件 II. 给定码字 $\mathbf{c}$, 计算重构向量 $\mathbf{x}'(\mathbf{c})$ 使其满足条件

$$\mathbf{x}'(\mathbf{c})=\frac{\int_{-\infty}^{\infty}d\mathbf{x}f_{\mathbf{X}}(\mathbf{x})\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))\mathbf{x}}{\int_{-\infty}^{\infty}d\mathbf{x}f_{\mathbf{X}}(\mathbf{x})\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))} \tag{9.23}$$

设置式(9.21)中的偏导数 $\partial D_1/\partial\mathbf{x}'(\mathbf{c})$ 为 0, 然后解出 $\mathbf{x}'(\mathbf{c})$ 可得式(9.23)。

图 9-5 描述的模型可作为图 9-6 描述的模型的特殊情形。具体地, 如果设置噪声 ν 的概率密度函数 $\pi(\nu)$ 等于 Dirac delta 函数 $\delta(\nu)$, 条件 I 和条件 II 分别归结为广义 Lloyd 算法的条件 1 和条件 2。

为了简化条件 I, 假定 $\pi(\nu)$ 为 ν 的光滑函数。可以证明式(9.22)定义的失真度量 D_2 的二阶近似包含两项(Luttrell,1989b):

- 常规失真项, 由平方误差失真 $\|\mathbf{x}-\mathbf{x}'(\mathbf{c})\|^2$ 定义。
- 由噪声模型 $\pi(\nu)$ 引起的曲率(curvature)项。

假设曲率项小, 对于图 9-6 的模型条件 I 可以近似为图 9-5 的无噪声模型的条件 1。这样又使条件 I 变成以前的最近邻编码规则。

至于条件 II, 可以利用随机下降学习实现它。具体地, 用因子 $\int d\mathbf{x}f_{\mathbf{X}}(\mathbf{x})$ 从输入空间 $\mathscr{X}$ 随机选择输入向量 $\mathbf{x}$, 并且更新重构向量 $\mathbf{x}'(\mathbf{c})$ 如下(Luttrell,1989b):

$$\mathbf{x}'_{new}(\mathbf{c})\leftarrow\mathbf{x}'_{old}(\mathbf{c})+\eta\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))[\mathbf{x}-\mathbf{x}'_{old}(\mathbf{c})] \tag{9.24}$$

其中 η 为学习率参数, $\mathbf{c}(\mathbf{x})$ 为条件 1 的最近邻编码近似。更新式(9.24)由检查式(9.21)的偏导数可得。这个更新应用于所有的 $\mathbf{c}$, 对此我们有

$$\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))>0 \tag{9.25}$$

可以认为式(9.24)描述的梯度下降过程为式(9.19)的失真度量 D_1 的一种最小化方法。也就是, 式(9.23)和(9.24)本质是同类型的, 区别在于式(9.23)为批处理方式的而(9.24)为连续的方式(即经过流的方式)。

458

更新式(9.24)等同于式(9.13)的(连续)SOM 算法, 记住在表 9-1 中所列的对应关系。因此, 可以说用于向量量化的广义 Lloyd 算法为具有 0 邻域大小的 SOM 算法的批处理训练模式; 对 0 邻域, $\pi(0)=1$。注意, 为了从 SOM 算法的批处理方式得到广义 Lloyd 算法我们无需作任何近似, 因为当邻域为 0 宽度时曲率项(和所有高阶项)不作任何贡献。

表 9-1 在 SOM 算法和图 9-6 的模型之间的对应

图 9-6 的编码 - 解码模型	SOM 算法
编码器 $\mathbf{c}(\mathbf{x})$	最佳匹配神经元 $i(\mathbf{x})$
重构向量 $\mathbf{x}'(\mathbf{c})$	突触权值向量 $\mathbf{w}_j$
概率密度函数 $\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))$	邻域函数 $h_{j,i(\mathbf{x})}$

下面给出讨论需注意的重要之处:

- SOM 算法为向量量化算法, 它提供输入空间 $\mathscr{X}$ 的良好近似。这个观点提供导出 SOM

算法的另一种途径，如式(9.24)的示例。

- 根据这个观点，SOM 算法中的邻域函数 $h_{j,i(\mathbf{x})}$ 有一个概率密度函数的形式。在 Luttrell (1991a)，考虑对图 9-6 的模型中噪声 ν 而言是合适的零均值高斯模型。因此我们对采用式(9.4)的高斯邻域函数又有了一个理论依据。

用求和作为对式(9.23)右端的分子和分母的积分的近似，批处理 SOM[8] 仅仅是式(9.23)的重写。注意在 SOM 算法的这种形式中，输入模式呈现给网络的顺序对特征映射的最终形式没有影响，且无需学习率调度。但算法仍需利用邻域函数。

性质 2　拓扑排序　*通过 SOM 算法计算的特征映射 Φ 是拓扑有序的，意味着网格中神经元的空间位置对应于输入模式的特定区域或特征。*

拓扑排序的特性[9]是更新公式(9.13)的直接结果，它使获胜神经元 $i(\mathbf{x})$ 的权值向量 $\mathbf{w}_i$ 移向输入向量 $\mathbf{x}$。它同样对距获胜神经元 $i(\mathbf{x})$ 近邻的神经元 j 的突触权值向量 $\mathbf{w}_j$ 的移动有作用。因此我们可以将特征映射 Φ 看成一个*弹性网或虚拟网*，它有在输出空间$\mathscr{A}$中描述的一维或两维的网格，并且它的节点具有权值作为输入空间$\mathscr{X}$中的坐标(Ritter, 1995)。因此算法的总的目标可以陈述如下：

459

指针或原型以突触权值向量 $\mathbf{w}_j$ 的形式逼近输入空间$\mathscr{X}$，使得特征映射 Φ 以这样一种方式提供根据某个准则而言表征输入向量 $\mathbf{x}\in\mathscr{X}$的重要特征的可信赖表示。

特征映射 Φ 通常在输入空间$\mathscr{X}$中显示，特别地，所有的指针(即突触权向量)显示为点，相邻神经元的指针按照网格的拓扑用线相连。因此，使用连线将两个指针 $\mathbf{w}_i$，$\mathbf{w}_j$ 连起来，表示相应神经元 i 和 j 在网格中是相邻神经元。

性质 3　密度匹配　*特征映射 Φ 反映输入分布在统计上的变化：在输入空间$\mathscr{X}$中样本向量 $\mathbf{x}$ 以高的概率抽取的区域映射到输出空间$\mathscr{A}$的更大区域，从而比在$\mathscr{X}$中样本向量 $\mathbf{x}$ 以低的概率抽取的区域有更好的分辨率。*

令 $f_{\mathbf{X}}(\mathbf{x})$ 表示随机输入向量 $\mathbf{x}$ 的多维 pdf(概率密度函数)。由定义，这个 pdf 在整个输入空间上的积分必须等于 1：

$$\int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\,d\mathbf{x} = 1$$

令 $m(\mathbf{x})$ 表示映射放大(magnification)因子，定义为输入空间$\mathscr{X}$的小体积 $d\mathbf{x}$ 中的神经元个数。放大因子在整个输入空间$\mathscr{X}$的积分一定等于网络中的神经元总数 l，即

$$\int_{-\infty}^{\infty} m(\mathbf{x})\,d\mathbf{x} = l \tag{9.26}$$

对于准确匹配输入密度的 SOM 算法，我们要求(Amari, 1980)

$$m(\mathbf{x}) \propto f_{\mathbf{X}}(\mathbf{x}) \tag{9.27}$$

这个性质意味着，如果输入空间中的一个特殊区域包含经常发生的刺激，那么与刺激出现较少的输入空间的区域相比，它将用特征映射中更大的区域表示。

一般地，在二维特征映射中放大因子 $m(\mathbf{x})$ 不能表示为输入向量 $\mathbf{x}$ 的概率密度函数 $f_{\mathbf{X}}(\mathbf{x})$ 的一个简单函数。只有在一维特征映射时才可能导出这样的关系。对这种特殊情况，我们发现与早些的推测(Kohonen, 1982)相反，它的放大因子 $m(\mathbf{x})$ 并不与 $f_{\mathbf{X}}(\mathbf{x})$ 成比例。基于采用的编码方法，在文献中报告了两个不同的结果：

1. 最小失真(畸变)编码，根据这个编码，式(9.22)的失真测度中的曲率项和高阶项由于噪声模型 $\pi(\mathbf{v})$ 仍然保留。这个编码方法可以产生结果

$$m(\mathbf{x}) \propto f_{\mathbf{X}}^{1/3}(\mathbf{x}) \tag{9.28}$$

这与标准的向量量化器得到的结果相同(Luttrell, 1991a)。

2. 最近邻编码，如同在 SOM 算法的标准形式中，它出现在忽略曲率项的时候。这个编码方法产生结果(Ritter, 1991)

$$m(\mathbf{x}) \propto f_{\mathbf{X}}^{2/3}(\mathbf{x}) \tag{9.29}$$

460

我们前面关于一族经常发生的刺激可以在特征映射中由更大的区域来表示的陈述仍然成立，虽然是用式(9.27)中描述的理想条件的失真形式。

作为一个一般规则(被计算机仿真确认)，由 SOM 算法计算的特征映射往往趋向于过高表示低输入密度区域和过低表示高输入密度区域。换句话说，SOM 算法不能为输入数据固有的概率分布提供可信赖的表示[10]。

性质 4　特征选择　*在具有非线性分布的输入空间中给定数据，自组织映射能够为逼近固有分布选择一组最好的特征。*

这个性质是性质 1 至性质 3 的自然结论。它使人想起前一章讨论的主分量分析的思想，但是如图 9-7 所示，它们有一个重要的区别。在图 9-7a 中展示被加性噪声损坏的线性输入 - 输出映射导出的零均值数据点的二维分布。这种情况下，主分量分析工作得很好：它告诉我们，在图 9-7a 中的"线性"分布的最好描述是定义成通过原点且平行于数据相关矩阵的最大特征值对应的特征向量平行的直线(即一维的"超平面")。接下去考虑图 9-7b 所描述的情况，这是受零均值加性噪声损坏的非线性输入 - 输出映射的结果。在这第二种情形从主分量分析计算的直线逼近不可能提供可接受的数据描述。另一方面，利用建立在一维神经元网格的自组织映射由于它的拓扑有序性质能够克服这个逼近问题。后一个逼近在图 9-7b 中说明。

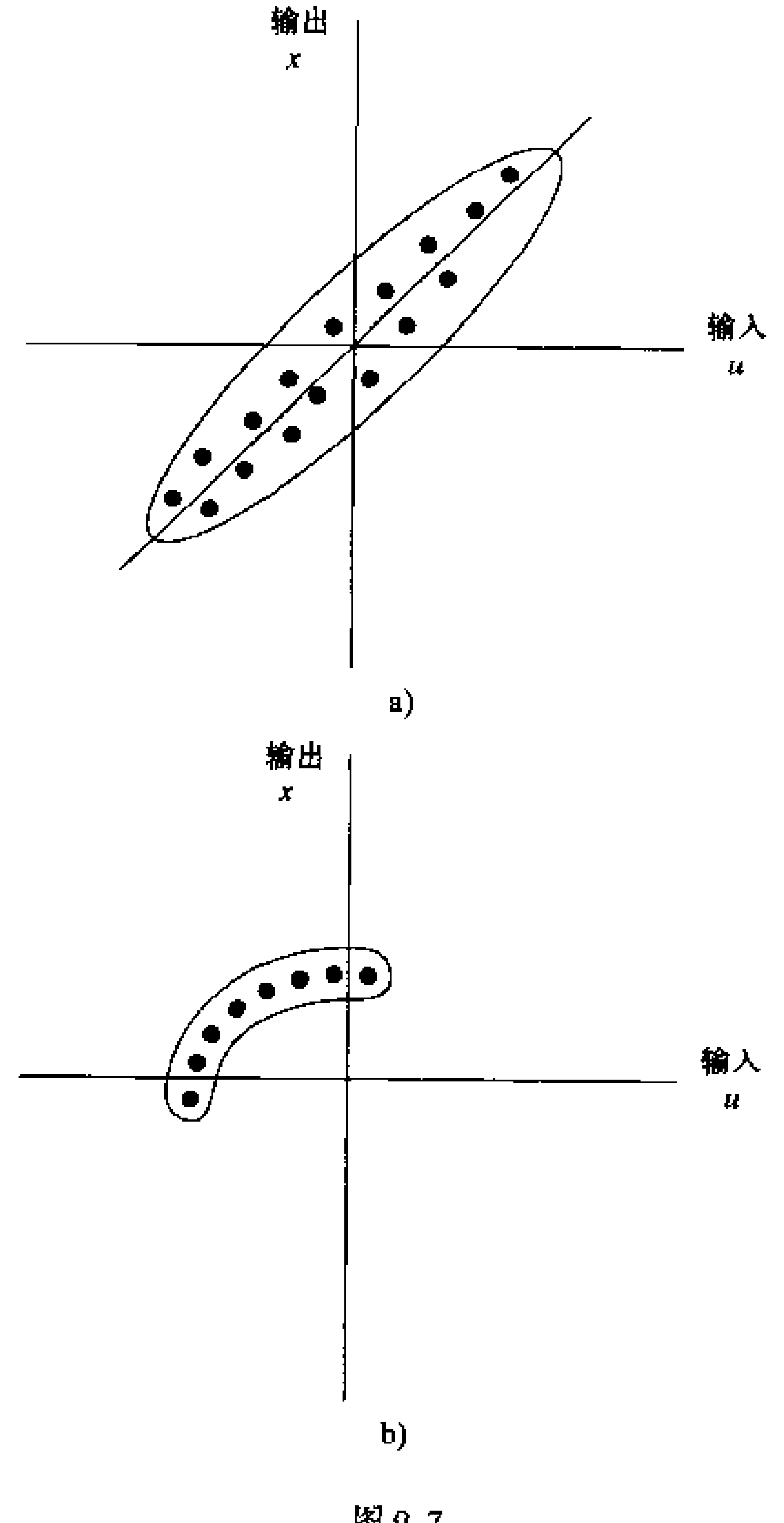


图 9-7

a)线性输入 - 输出映射产生的二维分布

b)非线性输入 - 输出映射产生的二维分布

精确地说，我们可以说自组织特征映射提供所谓*主曲线*[11](principal curve)或*主曲面*

(principal surface)的离散逼近(Hastie and Stuetzle,1989)，因此可以看成是主分量分析的非线性推广。

9.6 计算机仿真

由两维分布驱动的两维网格

我们使用计算机仿真来说明SOM算法的行为，通过研究100个神经元组成的网络，排列成10行和10列的两维网格。网络用二维输入向量 **x** 训练，它的分量 x_1 和 x_2 均匀分布在区域$\{(-1 < x_1 < +1);\ (-1 < x_2 < +1)\}$上。为了初始化网络，突触权值从一个随机集合抽取。

图9-8显示训练网络学习表示输入分布的三个阶段。图9-8a显示用来训练特征映射的 [461] 数据的均匀分布。图9-8b显示随机抽取的突触权值的初始值。图9-8c和图9-8d分别表示了在排序阶段和收敛阶段完成后突触权值向量的值，画出输入空间中点的图形。在图9-8中将网络中相邻神经元用线连起来(通过行和列)。

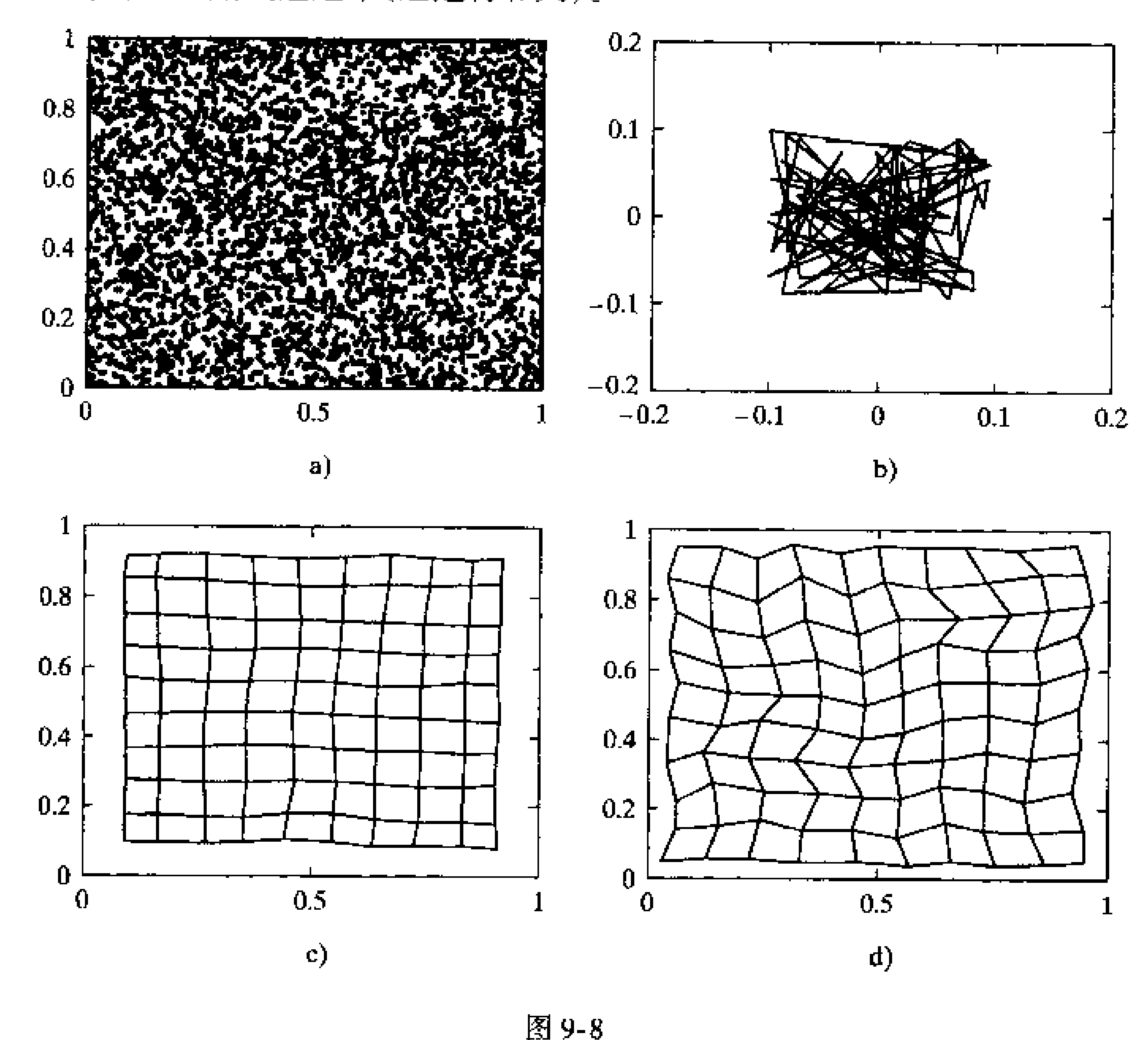


图9-8
a)输入数据分布 b)二维网格初始情况 c)排序阶段之后网格情况 d)收敛阶段之后网格情况

图9-8所示的结果展现表征SOM算法学习过程特点的排序阶段和收敛阶段。图9-8c显 [462] 示排序阶段，映射展开形成的网格。在这个阶段之后神经元映射为正确的排序。在收敛阶段映射散开充满输入空间。在第二阶段结束后，如图9-8d所示，映射中神经元的统计分布接

近输入向量的分布，除了一些边缘效果之外。比较图 9-8d 中特征映射的最终状态和图9-8a 的输入均匀分布，我们看出收敛阶段映射的调整抓住了可在输入分布中看到的局部不规则性。

SOM 算法的拓扑排序性质在图 9-8d 得到很好说明。尤其观察到算法(在收敛之后)抓住了输入中均匀分布的固有拓扑。图 9-8 所示的计算机仿真的输入空间$\mathscr{X}$和输出空间$\mathscr{A}$都是两维的。

由两维分布驱动的一维网格

我们现在考查当输入空间$\mathscr{X}$的维数大于输出空间$\mathscr{A}$的维数的情况。尽管不匹配，特征映射 Φ 常常能形成输入分布的拓扑表示。图 9-9 显示在特征映射演化过程中的三个不同的阶段，它的初始化如图 9-9b 所示，从矩形中抽取数据进行训练如图 9-9a 所示，但是，这次计算是在 100 个神经元的一维网格中进行的。图 9-9c 和图 9-9d 分别表示排序和收敛之后的特征映射。这里我们看到为了尽可能紧密地填充矩形从而提供二维输入空间$\mathscr{X}$的固有拓扑的良好近似，用算法计算的特征映射是非常失真的。在图 9-9d 所示的近似曲线类于 Peano 曲线(Peano curve)。以图 9-9 的特征映射为例的这种运算被称为维数削减(dimensionality reduction)，其中输入空间$\mathscr{X}$由将它投影到的低维输出空间$\mathscr{A}$来表示。 463

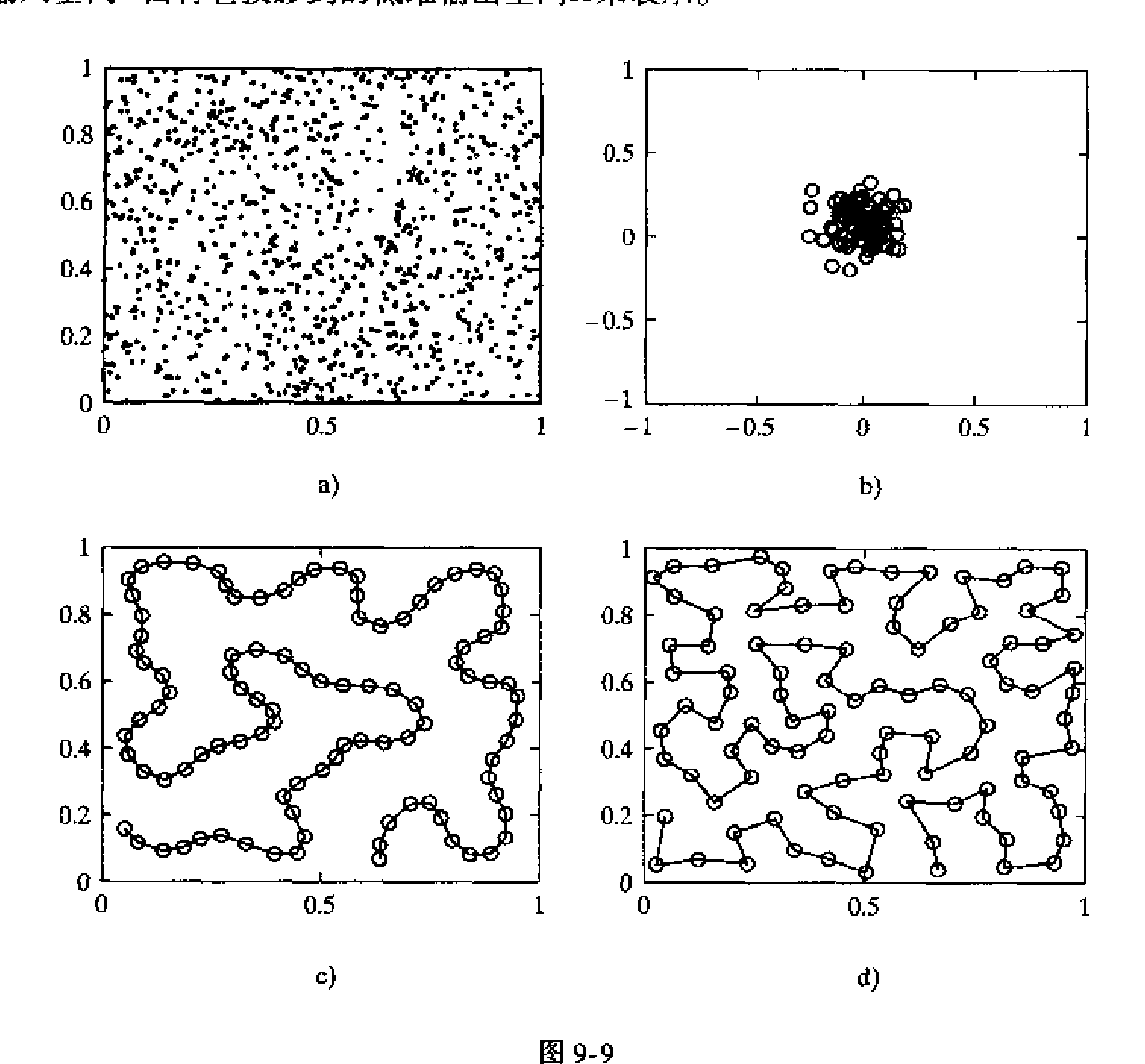


图 9-9

a)二维输入数据分布 b)一维网格初始情况 c)排序阶段之后的网格情况 d)收敛阶段之后的网格情况

仿真的参数设置

464

图 9-10 展示用于一维网格试验的邻域函数 $h_{j,i}(n)$和学习率参数 $\eta(n)$随时间(即回合次数)的变化。图 9-10a 所示的邻域函数参数 $\sigma(n)$开始时初始值 $\sigma_0=18$，然后在排序阶段的1000 次迭代中衰减到大约为 1。在同一阶段，学习率参数 $\eta(n)$开始时初始值 $\eta_0=0.1$，然后衰减到 0.037。图 9-10c 表示位于一维网格的中点的获胜神经元周围神经元的初始高斯分布。图 9-10d 显示在排序阶段结束后邻域函数的形状。在收敛阶段，学习率参数在 5000 步迭代中从 0.037 线性下降到 0.001。在同一阶段，邻域函数基本上减少到 0。

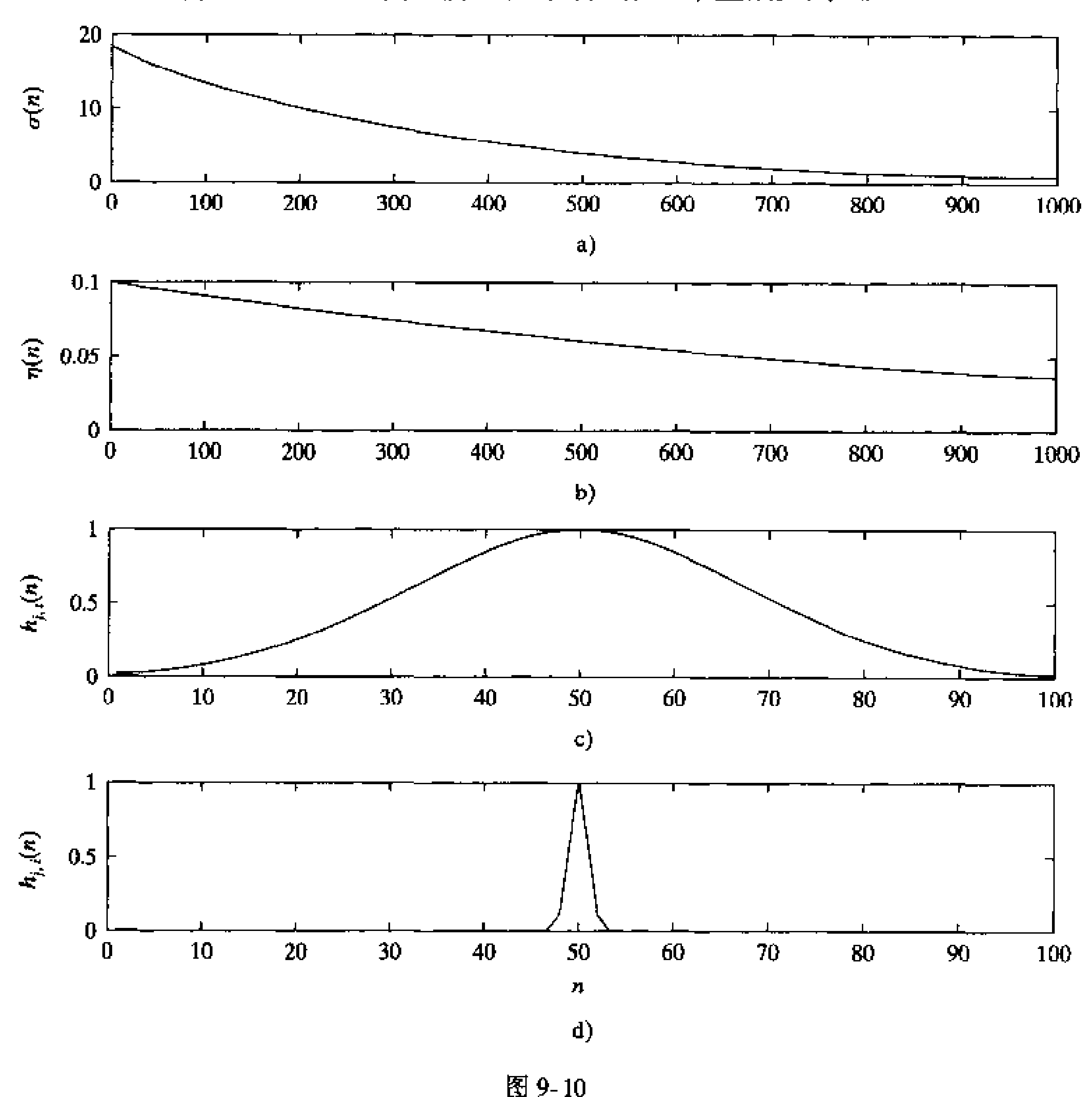


图 9-10

a)邻域函数参数 $\sigma(n)$呈指数衰减　b)学习率参数 $\eta(n)$的指数衰减　c)高斯邻域函数的初始形状
d)排序阶段结束后(即收敛阶段开始)邻域函数的形状

除了邻域函数是二维的外，图 9-8 涉及的二维网格的计算机仿真在排序阶段和收敛阶段的说明与一维网格的情况相似。参数 $\sigma(n)$从初始值 $\sigma_0=3$ 开始，然后在 1000 步迭代中减少到 0.75。图 9-11 显示在 10×10 的二维神经元网格中获胜神经元在点(7, 8)和 $\sigma_0=3$ 时二维高斯邻域函数 $h_{j,i}$的初始值。

465

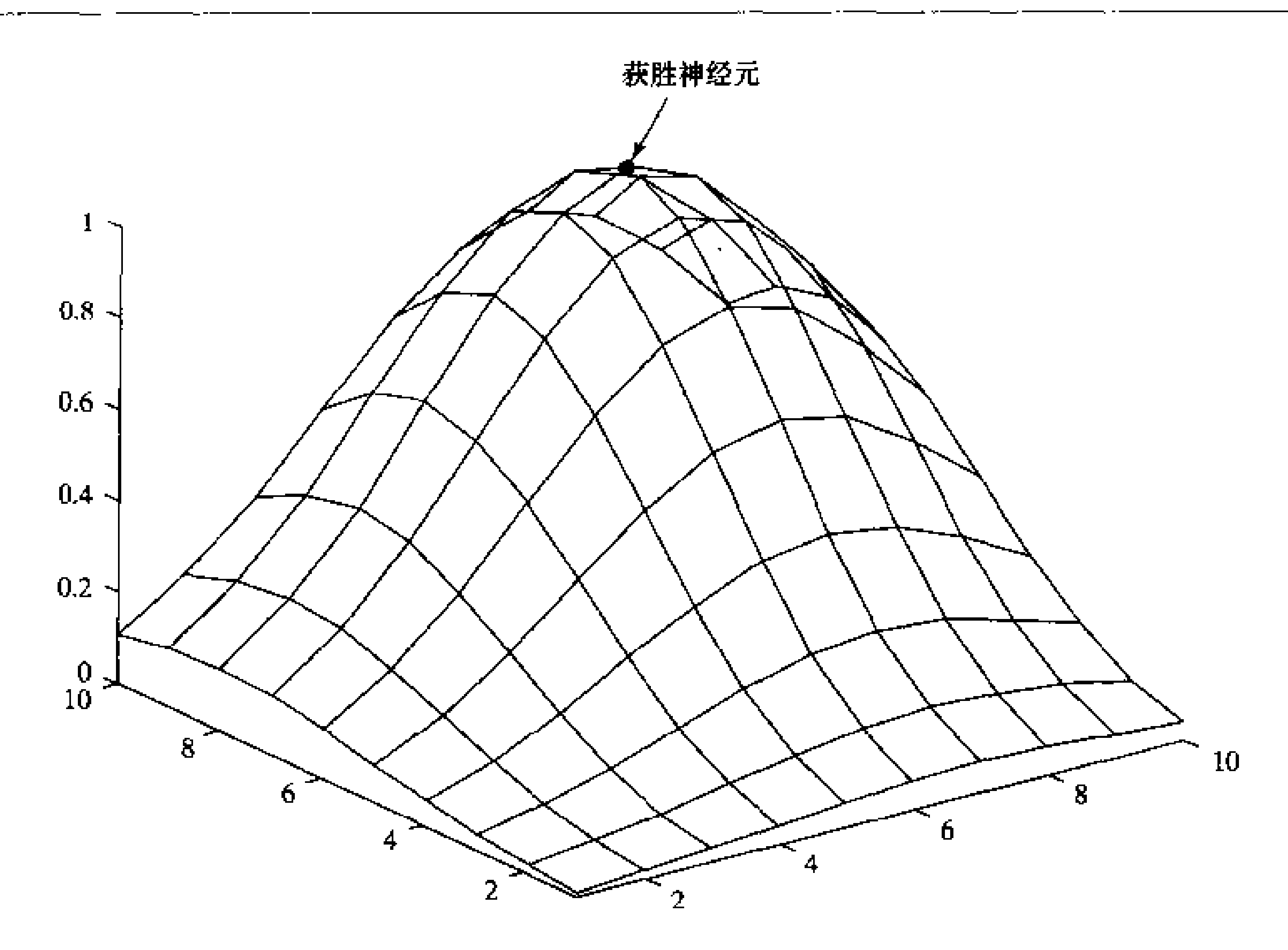


图 9-11 在 10×10 的二维神经元格形中获胜神经元在点(7,8)处的二维高斯邻域函数的初始情况

9.7 学习向量量化

在前面 9.6 节讨论的向量量化(vector quantization),是利用输入向量的固有结构进行数据压缩的技术(Gersho and Gray,1992)。具体地,输入空间被分成一些不同区域,并且对每一个区域定义一个重建向量。当一个新的输入向量提供给量化器时,首次确定向量所在的区域并且利用该区域的重构向量表示输入向量。这样,使用重建向量的编码替代原始输入向量来存储或传输,以一定的失真代价可实现在存储或传输带宽上的重大节省。可能的重构向量集被称作量化器的*码书*(code book),而它的成员被称为*码字*(code word)。

466

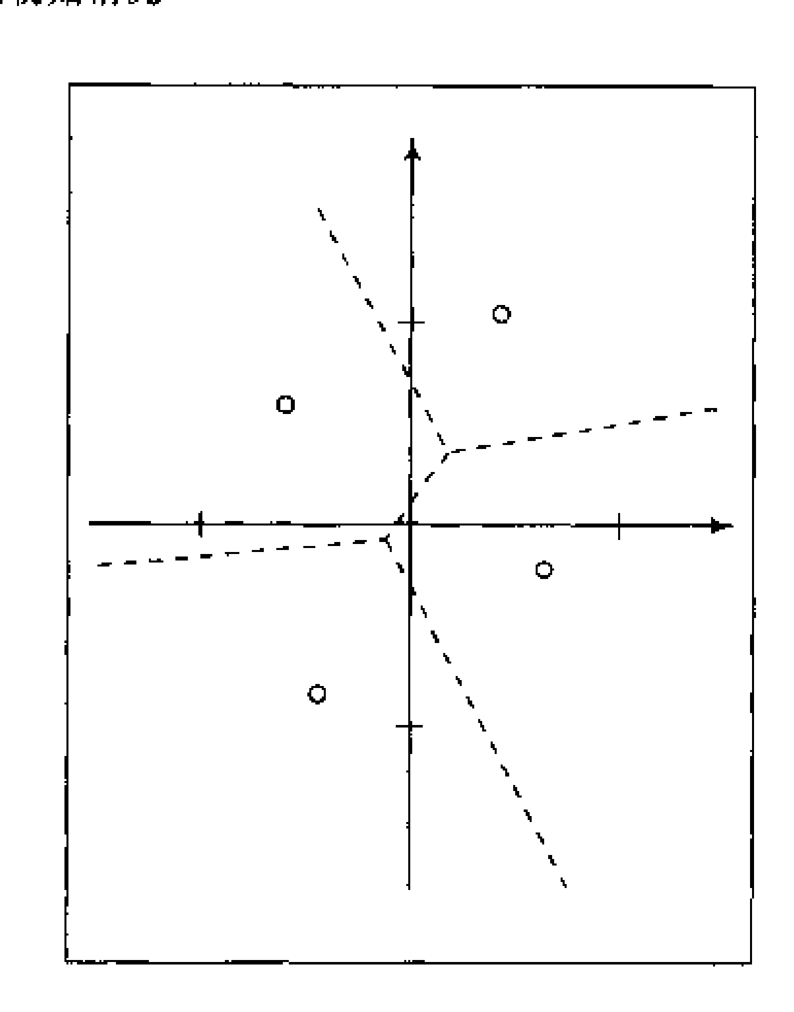


图 9-12 包含 4 个单元的 Voronoi 图(经 IEEE 许可,改自 R.M.Gray,1984)

一个有最小编码失真的向量量化器被称作 Voronoi *单元或最近邻域量化器*,因为关于输入空间点集的 Voronoi 单元对应于基于 Euclid 度量按最近邻规则对该空间的剖分(Gersho and Gray, 1992)。图 9-12显示一个输入空间分成四个 Voronoi 单元及它们相关的 Voronoi 向量(即重构向量)的例子。每个 Voronoi 单元包含输入空间中的那些点,它们在所有的点中最接近 Voronoi 向量。

SOM 算法提供一个无监督方式下计算 Voronoi 向量的逼近方法,其逼近通过特征映射中

神经元突触权值向量确定；这仅仅是在9.6节中讨论的SOM算法的性质1的重新陈述。如在图9-13所描绘的一样，特征映射的计算可以视为自适应解决模式分类问题两步中的第一步。第二步是学习向量量化，它提供一个最后细调特征映射的机制。

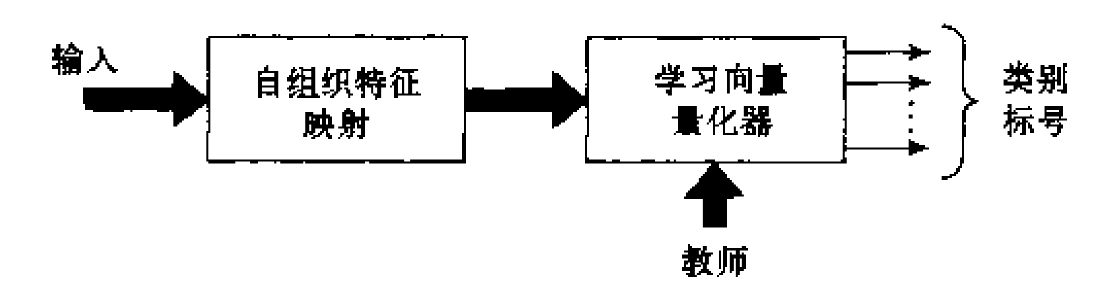


图9-13 利用自组织特征映射和学习向量量化器的自适应模式分类框图

学习向量量化器(learning vector quantization, LVQ)[12]是监督学习技巧，它使用分类消息来轻微移动Voronoi向量，以便提高分类器的决策区域质量。从输入空间随机抽取一个输入向量$\mathbf{x}$。如果输入向量$\mathbf{x}$的类别标号和Voronoi向量$\mathbf{w}$符合，Voronoi向量$\mathbf{w}$向输入向量$\mathbf{x}$的方向上移动。如果相反，输入向量$\mathbf{x}$的类别标号和Voronoi向量$\mathbf{w}$不符合，Voronoi向量$\mathbf{w}$向离开输入向量$\mathbf{x}$的方向移动。

467

设$\{\mathbf{w}_j\}_{j=1}^{l}$表示Voronoi向量集，$\{\mathbf{x}_i\}_{i=1}^{N}$表示输入(观察)向量集。假定输入向量多于Voronoi向量，在实际中这是典型的情况。学习向量量化(LVQ)算法如下：

(i)假定Voronoi向量$\mathbf{w}_c$距离输入向量$\mathbf{x}_i$最近。令$\mathscr{C}_{\mathbf{w}_c}$表示Voronoi向量$\mathbf{w}_c$的类别，$\mathscr{C}_{\mathbf{x}_i}$表示向量$\mathbf{x}_i$的类别标号。Voronoi向量$\mathbf{w}_c$调整如下：

- 如果$\mathscr{C}_{\mathbf{w}_c}=\mathscr{C}_{\mathbf{x}_i}$，则

$$\mathbf{w}_c(n+1)=\mathbf{w}_c(n)+\alpha_n[\mathbf{x}_i-\mathbf{w}_c(n)] \tag{9.30}$$

其中$0<\alpha_n<1$

- 相反，如果$\mathscr{C}_{\mathbf{w}_c}\neq\mathscr{C}_{\mathbf{x}_i}$，则

$$\mathbf{w}_c(n+1)=\mathbf{w}_c(n)-\alpha_n[\mathbf{x}_i-\mathbf{w}_c(n)] \tag{9.31}$$

(ii)其他Voronoi向量不作调整。

我们希望学习系数α_n随着迭代次数n的增加而递减。例如α_n初始值为0.1或更小，然后随着n线性递减。在通过输入数据几遍之后，Voronoi向量通常收敛并且训练完成。然而，如果应用方法不小心，可能会遇到困难。

9.8 计算机实验：自适应模式分类

在模式分类中，第一步和最重要的一步是特征选择(抽取)，它一般在无监督方式下完成。第一步的目标是选择小的合理特征集合，在其中(待分类的)输入数据的本质信息内容被集中起来。由于在9.5节讨论的自组织映射性质4，它适合特征选择的任务，尤其是当输入数据由非线性过程产生时。

模式识别的第二步是实际的分类，从输入数据选择特征赋予每个类。尽管自组织映射设计用来充当分类的角色，为了最好的性能建议对分类的第二步结合监督学习程序运行。自组织映射和监督学习模式的结合构成本质上混合的自适应模式分类的基础。

这种模式分类的混合方法可以采取不同的形式，取决于监督学习格式是怎样实现的。一

个简单的格式是使用前一节描述的学习向量量化器。这样我们有如图 9-13 所示的两步自适应模式分类器。

在这个实验里我们再次讨论标号 1(类$\mathscr{C}_1$)和标号 2(类$\mathscr{C}_2$)的部分重叠二维高斯分布模式的分类，在第 4 章里首次描述时它涉及用反向传播算法训练的多层感知器的应用。试验所用数据的散列图如图 4-13 所示。

图 9-14a 显示完成 SOM 算法训练后 5 × 5 神经元的二维特征映射。特征映射已被标定，根据对从输入分布中抽取的测试数据的响应每个神经元被指定为一个类或另一个类。图 9-14b显示由特征映射运行本身所实现的决策边界。 468

图 9-14c 显示利用 LVQ 以监督方式调整后的修正的特征映射。图 9-14d 显示 SOM 和 LVQ 算法联合产生的决策边界。比较这两个图以及它们在图 9-14a 和图 9-14b 对应部分，我们从量的方面看出利用 LVQ 所获得的效益。

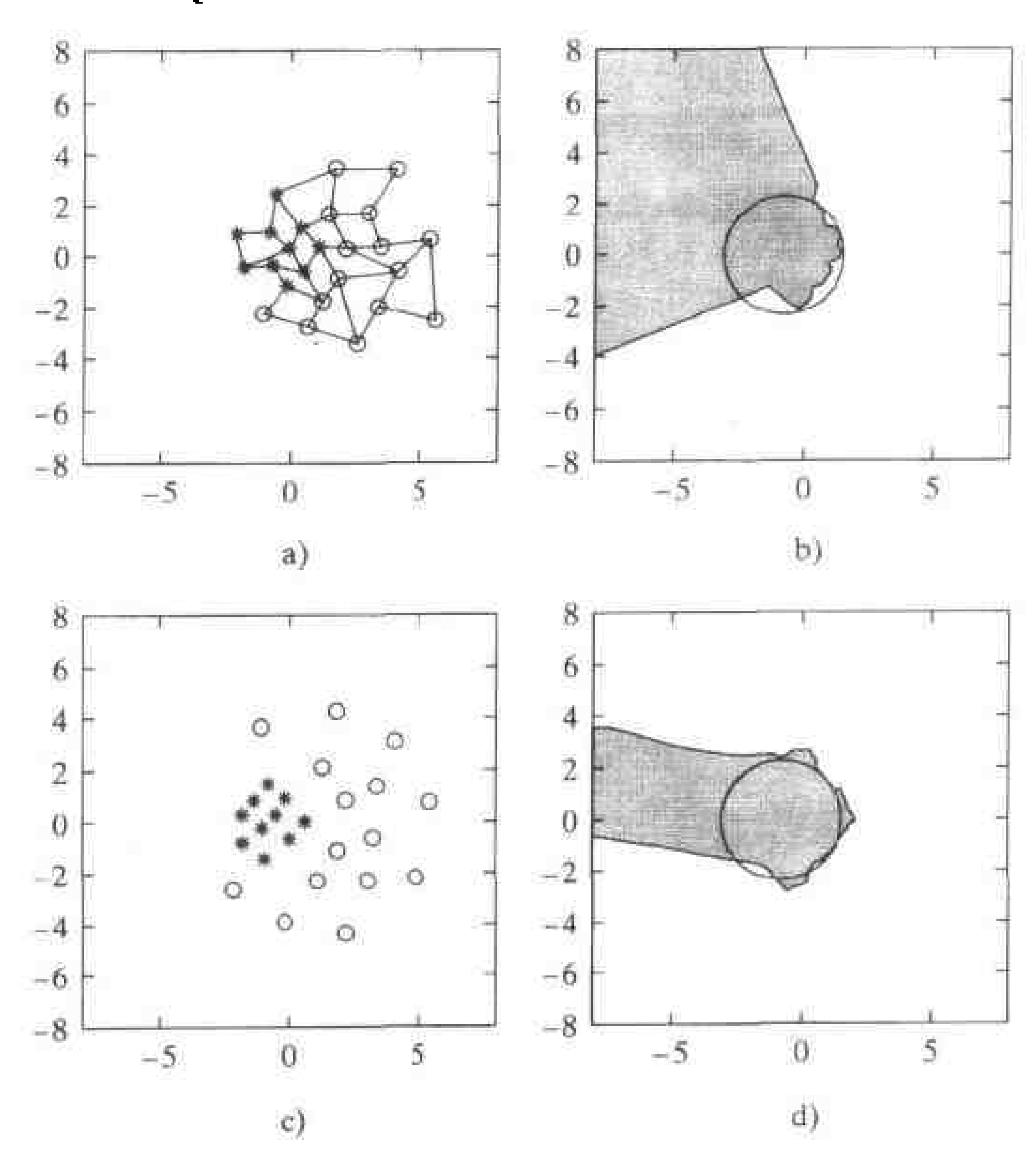


图 9-14

a)标定后的自组织映射 b)部分 a 的特征映射所建立的决策边界

c)学习向量量化后标定的映射 d)部分 c 的特征映射所建立的决策边界

表 9-2 给出特征映射自身和特征映射结合学习向量量化器的模式分类性能的小结。其中给出的结果为 10 次独立试验所得的结果，每次试验涉及使用 30 000 个模式作为测试数据。在每次试验中使用 LVQ 分类性能总有提高。特征映射本身的平均分类性能为 79.61%，而特征映射结合学习向量量化器的平均分类性能为 80.52%，这表示对特征映射本身有 0.91% 的提高。作为参考框架，我们回想这个试验的最优 Bayes 分类器性能为 81.51%。 469

表 9-2 对用 5×5 网格的二维重叠高斯分布的计算机试验分类性能(百分比)小结

试验	特征映射自身	特征映射和学习向量量化器串联结合
1	79.05	80.18
2	79.79	80.56
3	79.41	81.17
4	79.38	79.84
5	80.30	80.43
6	79.55	80.36
7	79.79	80.86
8	78.48	80.21
9	80.00	80.51
10	80.32	81.06
平均	79.61%	80.52%

9.9 分层向量量化

在9.6节自组织特征映射的性质1的讨论中，我们指出在向量量化方面它与广义 Lloyd 算法紧密相关。向量量化是有损(lossy)数据压缩的一种形式，有损的意思是指一些包含在输入数据中的信息由于压缩的结果丢失了。数据压缩植根于 Shannon 信息论的一个分支，称为*率失真*(rate distortion)*理论*(Cover and Thomas,1991)。为了目前处理的分层向量量化的目的，以陈述下面率失真理论的基本结果作为开始是很适合的(Gray,1984)：

> *通过获得向量编码而不是标量编码，总是能够取得好的数据压缩性能，即使数据源是无记忆的(例如，它提供一系列独立随机变量)，或者数据压缩系统有记忆的(即编码器的动作依赖于编码器以前的输入或输出)。*

470 这个基本结果构成对向量量化作出贡献的广泛研究工作的基础(Gersho and Gray,1992)。

然而，传统的向量量化算法要求大量的计算，这妨碍了它们的实际使用。向量量化最费时的部分是编码操作。为了编码过程，输入向量必须与每一个在码书中的码字向量作比较，以便决定哪一个特别的码字产生最小失真度。例如对于码书包含 N 个码向量，编码所花的时间依赖于 N 的阶，这样对大的 N 值所花时间就多。在 Luttrell(1989a)描述一个*多阶段分层*(multistage hierarchical)*向量量化器*，它用编码速度换取精度。这个模式不是标准的码书的树搜寻；它是真正新的。多阶段分层向量量化器试图将所有的向量量化过程分解成许多子操作，每个子操作仅要求少量的计算。理想的分解对每个子操作简化为简单的查表。通过巧妙地使用 SOM 算法来训练量化器的每一阶段，准确性的丢失可能很少(低到几分之一分贝(decibel))，同时计算速度的增益可能很大。

考虑两个向量量化器 VQ_1 和 VQ_2，其中 VQ_1 将它的输出送到 VQ_2 作为其输入。VQ_2 的输出是应用于 VQ_1 的原输入信号的最终编码形式。在运行它的量化过程中，VQ_2 不可避免地抛弃一些信息。就 VQ_1 而言，VQ_2 仅有的作用是扭曲 VQ_1 输出的信息。这样很明显对 VQ_1 的正确的训练方法是 SOM 算法，它说明 VQ_2 诱导的信号失真(Luttrell,1989a)。为了使用广义 Lloyd 算法来训练 VQ_2，我们仅需要假定 VQ_2 的输出在重建之前没有被损坏。从而我们无需引入噪声模型(在 VQ_2 的输出)及相应的有限宽度邻域函数。

我们可以推广这个启发式的结论到多阶段量化器。必须设计每一阶段使之考虑所有的后面阶段导致的失真并且为它建立噪声模型。为这样做，使用 SOM 算法训练量化器的所有阶段，除了最后一个阶段适宜用广义的 Lloyd 算法训练。

分层向量量化过程是多阶段向量量化的特例(Luttrell, 1989a)。作为一种例证，考虑 4×1 的输入向量 $\mathbf{x}=[x_1,x_2,x_3,x_4]^T$ 的量化。在图 9-15a 我们给出用于 $\mathbf{x}$ 的单阶段向量量化器。另外，我们可以使用如图 9-15b所描绘的两阶段分层量化器。这两个模式的重要区别是在图 9-15a 的量化器输入维数为 4 而在图 9-15b 中它是 2。因此，图 9-15b 的量化器要求小规模的查用表，因此比图 9-15a 的量化器实现简单。这是分层量化器比传统量化器优越之处。

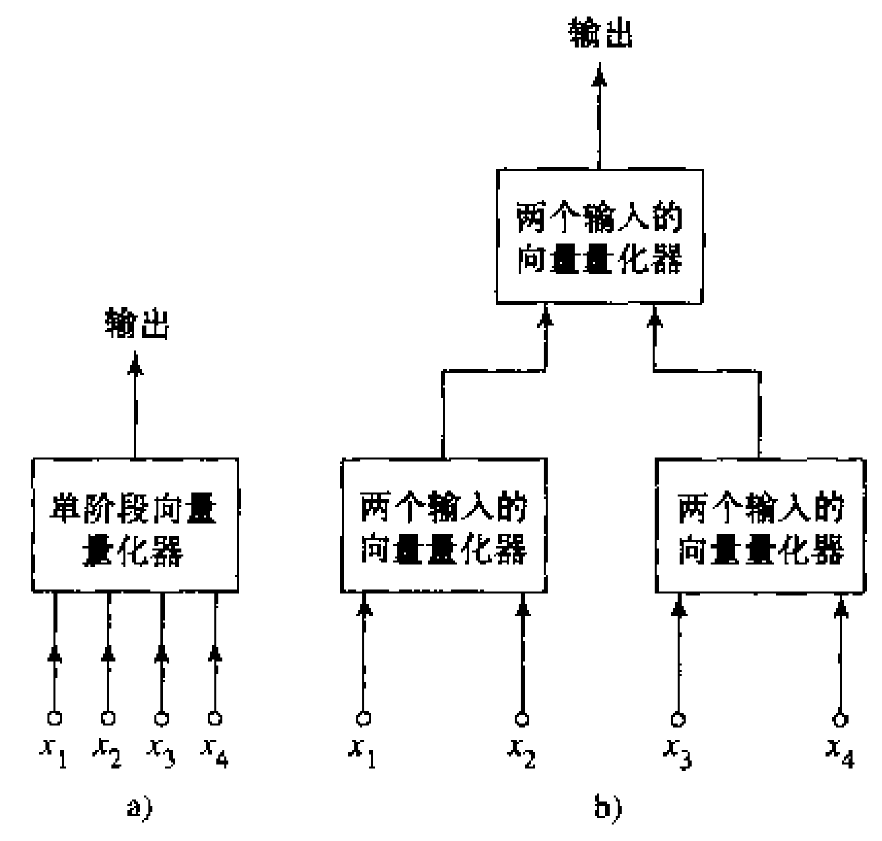


图 9-15
a)具有四个输入的单阶段向量量化器
b)使用两个输入的两阶段分层向量量化器
(摘自 S.P. Luttrell(1989a), British Crown 版权)

Luttrell(1989a)展示多阶段分层向量量化器应用到不同的随机时间序列的性能，编码准确度丢失很少。在图 9-16 重新产生了 Luttrell 的结果，它是利用一阶自回归(first-order autoregressive, AR)模型：

$$x(n+1)=\rho x(n)+\nu(n) \tag{9.32}$$

产生的，具有高斯噪声过程，其中 ρ 为 AR 系数，$\nu(n)$为独立同分布(idd)的高斯随机变量，具有零均值和单位方差。因此我们可以证明 $x(n)$的特征如下：

$$E[x(n)]=0 \tag{9.33}$$

471

$$E[x^2(n)]=\frac{1}{1-\rho^2} \tag{9.34}$$

$$\frac{E[x(n+1)x(n)]}{E[x^2(n)]}=\rho \tag{9.35}$$

因此 ρ 也可看成时间序列$\{x(n)\}$的相关系数。为了按照式(9.32)初始化时间序列的生成，对 $x(0)$使用均值为零和方差为 $1/(1-\rho^2)$的高斯随机变量，并且相关系数使用 $\rho=0.85$。

对于向量量化使用类似于图 9-15b 中的二分树一样具有四维输入空间的分层编码器。对于 AR 时间序列$\{x(n)\}$，平移对称意味着仅需两个不同的查用表(look-up table)。每张表的大小按指数依赖于输入比特数，而线性依赖于输出比特数。在训练过程中，需要大量比特数表示式(9.24)描述的更新的正确计算的数；这样在训练期间不使用查用表。但是一旦训练完成，比特数可降低至它们的正常水平，并且按要求填充表项。对于如图 9-15b 显示的编码器，每个输入样本用 4 比特近似。对解码器的各个阶段，使用 $N(=17)$个码字向量，这样从每个查用表的输出比特数也近似为 4。因此第一阶段和第二阶段的查用表的地址空间的大小为 $256(=2^{4+4})$，这意味着查用表的表示所需存储要求是适中的。

图 9-16 显示用 $x(n)$作为输入得到的编码－解码结果。图 9-16a 的下半部分显示两阶段

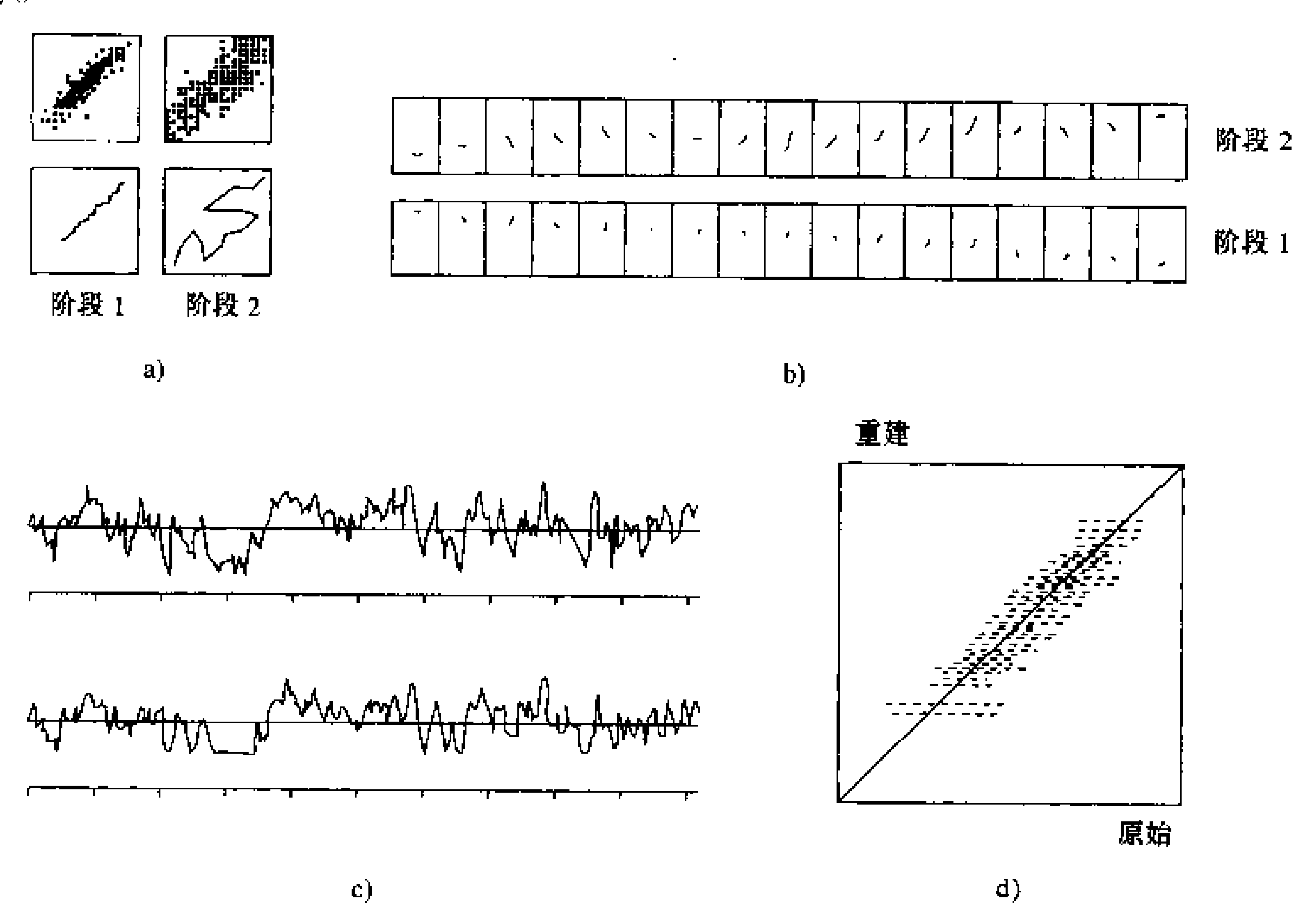
中每个阶段的码字向量为一条嵌入二维输入空间的曲线；图 9-16a 的上半部分表示相应的用 16×16 比特的共生(co-occurrence)矩阵的估计。图 9-16b 表示如下时间序列片段：

- 出第一个编码阶段计算的码字向量。
- 保持其他变量固定，由第二阶段最小化均值平方失真计算出的重构向量。

472

图 9-16c 显示 512 个样本，包括原始时间序列(顶部曲线)和从最后一个编码器阶段的输出得到的它的重构(底部曲线)；图 9-16c 的水平方向的刻度是图 9-16b 的一半。最后，图 9-16b 表示从一对样本(原始时间序列样本和它的相应重构)产生的共生矩阵。图 9-16d 中的带宽指示由分层向量量化产生的失真程度。

检查图 9-16c 的波形，看出重构对原始时间序列是好的表示，除了一些正和负的峰值被剪除。根据 Luttrell(1989a)归整化后的均值平方失真经计算为 0.15，它同每个样本用一比特的单阶段 4 – 样本块编码器所获得的 8.8 分贝几乎一样好(0.5 分贝的损失)(Jayant and Noll, 1984)。



473

图 9-16 用于相关高斯噪声输入的两阶段编码/解码结果
相关系数 $\rho=0.85$(摘自 S.P.Luttrell(1989a)，British Crown 版权)

9.10 上下文映射

自组织特征映射有两种明显不同的可视化方法。在一种可视化方法中，特征映射被视为有弹性的网络，此时向量权值被视为对应神经元的指针，指向输入空间。这种可视化方法特别适用于显示 SOM 算法的拓扑排序属性，如 9.6 节给出的计算机仿真实验结果所说明。

在第二种可视化方法中，对两维网格(表示网络的输出层)的神经元赋予类别标号，它取决于每个测试模式(以前未见过)怎样激活自组织网络中的特定神经元。作为仿真第二阶段的结果，两维网格中的神经元被剖分成许多相干区域(coherent region)，相干的含义是神经元每

个分组表示邻接符号或标号的一个独特的集合(Ritter and Kohonen, 1989)。这里假定第一步产生良序的特征映射的正确条件成立。

例如，考虑表 9-3 中给出的数据集合，它们是关于许多不同动物的。表的每一列是对动物的示意性描述，它是根据左边 13 个不同的属性的出现(= 1)或不出现(= 0)。一些属性例如"羽毛"和"两条腿"是相关的，而其他许多属性是不相关的。对表头给出的每个动物，它的属性代码 $\mathbf{x}_a$ 是由 13 个属性构成。动物本身由符号代码 $\mathbf{x}_s$ 指定，符号代码的组成必须不表达动物的任何信息或它们之间已知的相似点。例如当前的例子，$\mathbf{x}_s$ 是由一个列向量构成，它的第 k 个元素，表示动物 $k = 1,2,\cdots,16$，赋予一个固定值 a；剩下的元素都置成 0。参数 a 与属性代码比较而言决定符号代码之间的相关影响。为了确定属性代码是重要的一个，a 选择为 0.2。每个动物的输入向量 $\mathbf{x}$ 是一个 29 个元素的向量，表示属性代码 $\mathbf{x}_a$ 和符号代码 $\mathbf{x}_s$ 的联合，表示为

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_a \end{bmatrix} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{x}_a \end{bmatrix}$$

最后，每个数据向量都被归一化为单元长度。这样产生的数据集的模式被呈现给 10×10 的两维神经元网格，神经元的权值按照 9.4 节中阐述的 SOM 算法调整。训练连续进行 2000 次迭代，此时特征映射应该达到一个稳定状态。接着，由一个动物包含的符号代码 $\mathbf{x} = [\mathbf{x}_s, \mathbf{0}]^T$ 定义的测试模式呈现给自组织网络，并且确定具有最强响应的神经元。对所有的 16 种动物都重复这样做。

表 9-3 动物的名称和它们的属性

动物		鸽	母鸡	鸭	鹅	猫头鹰	隼	鹰	狐狸	狗	狼	猫	虎	狮	马	斑马	母牛
为	小型	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	中型	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	大型	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
有	2条腿	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4条腿	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	毛发	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	蹄	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	鬃	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	羽毛	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
善长	猎食	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	奔跑	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	飞翔	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	游泳	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

474

按刚才陈述的方法处理，我们得到图 9-17 所示的映射，其中标定名称的神经元代表它们对各自的测试模式有最强的响应，点代表有较弱的响应的神经元。

图 9-18 对相同的自组织网络显示"模拟电极渗透映射"的结果。但是，图中网络的每个神经元用使之产生最好响应的特定动物名称标记。图 9-18 清楚地表明在 16 个不同的动物中特征映射能抓住"种属关系"。这里有三个不同的聚类，一个表示"鸟类"，第二个表示"平和的种属"，第三个表示"猎手"。

图 9-18 表示的特征映射类型称为上下文映射或语义映射(Ritter and Kohonen, 1989;

dog	·	·	fox	·	·	cat	·	·	eagle
·	·	·	·	·	·	·	·	·	·
·	·	·	·	·	·	·	·	·	owl
·	·	·	·	·	·	tiger	·	·	·
wolf	·	·	·	·	·	·	·	·	hawk
·	·	·	lion	·	·	·	·	·	·
·	·	·	·	·	·	·	·	·	dove
horse	·	·	·	·	·	·	hen	·	·
·	·	·	·	cow	·	·	·	·	goose
zebra	·	·	·	·	·	·	duck	·	·

图 9-17 包含对它们各自输入具有最强响应的标定神经元的特征映射

475

dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
dog	dog	fox	fox	fox	cat	cat	cat	eagle	eagle
wolf	wolf	wolf	fox	cat	tiger	tiger	tiger	owl	owl
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	tiger	tiger	tiger	hawk	hawk
wolf	wolf	lion	lion	lion	owl	dove	hawk	dove	dove
horse	horse	lion	lion	lion	dove	hen	hen	dove	dove
horse	horse	zebra	cow	cow	cow	hen	hen	dove	dove
zebra	zebra	zebra	cow	cow	cow	hen	hen	duck	goose
zebra	zebra	zebra	cow	cow	cow	duck	duck	duck	goose

图 9-18 利用"模拟电极渗透映射"的语义映射。映射被分成三个不同区域，分别代表鸟类、平和种属及"猎手"

Kohonen,1997a)。这个映射与大脑皮质的映射相似(即在大脑皮质里形成的计算映射)，这在9.2节里作了简要讨论。作为利用SOM算法产生的结果，上下文映射在众多领域找到了应用，诸如文本的音素类别的无监督分类，遥感(Kohonen, 1997a)，数据探测或数据挖掘(Kohonen,1997b)。

9.11 小结和讨论

由Kohonen(1982)提出的自组织映射是一个巧妙的神经元网络，它建立在一维或两维的神经元网格上，用于捕获包含在输入(数据)空间中感兴趣的特征。为此，它利用神经元权值向量作为原型提供一个输入数据的结构表示。SOM算法受到神经生物学的激发，综合第8章中讨论的所有自组织的基本机制：竞争、合作和自放大。因此它可以作为虽退化但一般的模型，描述在复杂系统中从完全混乱开始最终出现整体有序的现象。

自组织映射也可以被看作向量量化器，从而提供一个导出调整权值向量的更新规则的原理性方法(Luttrell,1989b)。后一种方法明确地强调邻域函数作为概率密度函数的作用。

然而应该指出，基于使用在式(9.19)中的平均分布 D_1 作为极小化代价函数的后一种方法，仅当特征映射被很好的排序后才是合理的。在Erwin et al.(1992b)中，证明在自适应过程的排序阶段(即在初始是高度混乱的特征映射的拓扑排序期间)自组织映射的学习动态系统不能用一个代价函数的随机梯度下降描述。但就一维网格的情况来说，它可以用一组代价函数描述，对于网络中每个神经元，一个对应的代价函数随随机梯度下降独立地被最小化。

476

关于Kohonen的SOM算法，令人惊奇的是它的实现如此简单，但在一般设置下分析它的

性质数学上却如此困难。虽然几个研究者使用相当有力的方法来分析它，但是，他们仅获得有限的应用性结果。在 Cottrell et al.(1997)中给出关于 SOM 算法理论方面的结果的综述。尤其最近由 Forte and Pagés(1995,1997)得出的结果引人注目，结果表明就一维网格情况而言，可严格证明：在自组织阶段结束后，SOM 算法"几乎确定"收敛到一个惟一状态。这个重要的结果已被证明对一大类邻域函数成立。然而，在多维情况下尚未得到同样的结论。

最后一点疑问是自然的。既然自组织特征映射是由大脑皮质映射的思想所激发的，很自然会问是否这种模型可以实际解释皮质映射的形成。Erwin et al.(1995)进行了这项研究。他们发现自组织特征映射可以解释猕猴初级视觉皮质中计算映射的形成。这项研究的输入空间的维数是 5 维：两维为视觉空间接收域的位置，剩下的三维代表方向优先、方位选择和视觉优势。皮质表面被分成小块，每块被视为两维网格的计算单元(即人工神经元)。在一定假设下，表明 Hebb 学习导致空间模式的定位和视觉优势与在猕猴中发现的非常相似。

注释和参考文献

[1] 图 9-1 的两个特征映射模型是由 von der Malsburg(1973)的自组织的先驱性研究所激发，Malsburg 注意到视觉皮质的模型不能整体被基因预先确定；相反涉及突触学习的自组织过程可能导致特征敏感的皮质细胞的局部排序，但是在 von der Malsburg 的模型中不能取得全局拓扑序，因为模型使用固定的(很小的)邻域，von der Malsburg 的计算机仿真也许是第一次展示自组织。

[2] Amari(1980)在某种程度上放松对后突触神经元的突触权值的限制。Amari 给出的数据分析阐明由自组织形成的皮质映射的动态稳定性。

[3] Kohonen(1993,1997a)讨论自组织映射的神经生物学的可行性。

[4] Grossberg(1969b)在神经网络文献中第一次引入式(9.3)描述的竞争学习规则。

[5] 在 Kohonen(1982)导出的 SOM 算法的原始形式中，拓扑邻域假定为有固定的范围。令 $d_{j,i}$ 表示在邻域函数内获胜神经元 i 和兴奋神经元 j 的侧向距离，一维网格情形的拓扑邻域定义为

$$h_{j,i} = \begin{cases} 1, & -K \leqslant d_{j,i} \leqslant K \\ 0, & \text{其他} \end{cases} \tag{1}$$

其中 $2K$ 为兴奋神经元一维邻域的总长度。与神经生物学考虑相反，式(1)描述的模型意味着在拓扑邻域内所有神经元以相同的速度点火，且这些神经元内部的相互作用与它们到获胜神经元的侧向距离无关。

[6] Erwin et al.(1992b)表明当 SOM 算法利用非凸的邻域函数时会出现亚稳定状态，它表示在特征映射设置中的拓扑缺陷。Gauss 函数是凸的而矩形函数不是凸函数。一个宽的邻域函数，如宽 Gauss 函数，形成拓扑排序的时间比非凸邻域函数(如矩形函数)所花的时间短，这是因为没有亚稳定状态。 477

[7] 在通信和信息论的文献中，提出了著名的标量量化的早期方法，即 Lloyd 算法。这个算法首先由 Lloyd 在 Bell 实验室 1957 年未发表的报告中描述(Lloyd,1957)，很久以后才发表(Lloyd,1982)。Lloyd 算法有时也称为"最大量化器"。用于向量量化的广义 Lloyd 算法(generalized Lloyd algorithm,GLA)是 Lloyd 算法的直接推广。广义 Lloyd 算法在 McQueen(1967)将其作为统计聚类的工具之后有时称为 *k* - 均值算法。在 Linde et al.(1980)之后

的数据压缩文献中它有时也称为 LBG 算法。Lloyd 算法及广义 Lloyd 算法的历史评述可参看 Gersho and Gray(1992)。

[8] Kohonen(1993)给出的实验结果表明，SOM 算法的集中方式比它的在线方式快。但是使用集中方式时 SOM 算法失去自适应能力。

[9] 自组织映射的拓扑性质可由不同方法定量评价。一种这样的定量度量称为*地形图产品*(topographic product)，它在 Bauer and Pawelzik(1992)中描述，它可用于比较属于不同维数的不同特征映射的真实行为，但是只有当网格维数和输入空间维数匹配这种度量才是量化的。

[10] SOM 算法无能力提供输入数据的固有分布的可信表示，这一点促使对算法的修正和真实表示输入的新自组织算法的发展。

在文献中有两类 SOM 算法修正的报导。

(i)*修改竞争过程*。DeSieno(1988)在网格中用记忆形式跟踪单个神经元累计激活量。具体地，添加"良心"机制影响 SOM 算法的竞争过程。这样做使得每个神经元不管它在网格中的位置如何都有机会以接近于理想值 $1/l$ 的概率获胜，其中 l 为总的神经元数。习题 9.8 给出具有良心机制的 SOM 算法描述。

(ii)*修改自适应过程*。在这第二种方法中，对用于调整邻域函数内每个神经元权值向量的更新规则进行修改，控制特征映射的放大性质。在 Bauer et al. (1996)中，表明通过对更新规则添加可调步长参数，可以为特征映射提供输入数据的可信表示。Lin et al.(1997)遵循相似的途径引入 SOM 算法的两种修改：

- 修改更新规则，抽取输入向量 $\mathbf{x}$ 和问题中神经元 j 的权值向量 $\mathbf{w}_j$ 的直接依赖性。
- 利用为可分输入分布特别设计的等变化(equivariant)剖分替代 Voronoi 剖分。

478

这第二种修改使得 SOM 算法能进行盲源分离。(盲源分离在第 1 章有简单讨论，在第 10 章作详细讨论。)

所提到的修改建立在标准 SOM 算法的这种或那种形式上。Linsker(1989b)采用一种完全不同方法。具体地，利用最大化输出信号和带加性噪声的输入信号之间的互信息的方法，导出用于地形图映射形成的全局学习规则。(植根于 Shannon 信息论的互信息的定义在第 10 章讨论。)Linsker 的模型产生与输入分布精确匹配的神经元分布。利用信息论的方法以自组织方式处理地形图映射形成也在 Van Hulle(1996,1997)中讨论。

[11] SOM 算法和主曲线之间的关系在 Ritter(1992)以及 Cherkassky and Mulier(1995)讨论。寻找主曲线的算法分为两步(Hastie and Stuetzl,1989)：

1. *投影*。对每个数据点寻找在曲线上的最近投影或最接近点。
2. *条件期望*。应用散列图沿曲线长度平滑投影值。推荐的程序是从大范围开始平滑然后逐渐减少。

这两步和向量量化及 SOM 算法所进行的邻域退火相似。

[12] 1986 年 Kohonen 提出了学习向量量化的思想，Kohonen(1990b,1997a)描述这个算法的 3 种形式。在 9.7 节讨论的算法形式是学习向量量化的第一种，Kohonen 称之为 LVQ1。

学习向量量化算法是随机逼近算法。Baras and La Vigna(1990)用第 8 章叙述的常微分方程(ODE)方法讨论这个算法的收敛性质。

习题

SOM 算法

9.1 函数 $g(y_j)$ 表示响应 y_j 的非线性函数，它如同在(9.9)中那样用于 SOM 算法。如果 $g(y_j)$ 的 Taylor 展示的常数项不为零，讨论这会产生什么结果？

9.2 假设 $\pi(\nu)$ 为图 9-6 模型的噪声 ν 的光滑函数，利用式(9.9)的失真度量的 Taylor 展开，确定噪声模型 $\pi(\nu)$ 导致的曲率项。

9.3 有时说 SOM 算法保持输入空间中存在的拓扑关系。严格地说，这种性质只有输入空间的维数与神经元网格的维数相等或再低时才能保证。讨论这个陈述的正确性。

9.4 一般说基于竞争学习的 SOM 算法对硬件故障不具有容错性，但是算法对输入的小的扰动引起输出从获胜神经元跳到相邻的神经元具有容错性。讨论这两个陈述的含义。

9.5 考虑由(9.23)表示的 SOM 算法的离散形式获得的集中方式，表示为

$$\mathbf{w}_j = \frac{\sum_i \pi_{j,i}\mathbf{x}_i}{\sum_i \pi_{j,i}}, j = 1,2,\cdots,l$$

479

证明 SOM 算法的这种形式可以表示成和 Nadaraya-Watson 回归估计器相似的形式(Cherkassky and Mulier, 1995)；这个估计器在第 5 章讨论。

学习向量量化

9.6 在本题中考虑 9.7 节的学习向量量化算法的优化形式(Kohonen, 1997a)。我们希望调整在不同时间对 Voronoi 向量所做的修正效果使得参照学习周期结束时有相同影响。

(a)首先，证明式(9.30)和(9.31)可集成为一个等式

$$\mathbf{w}_c(n+1) = (1 - s_n\alpha_n)\mathbf{w}_c(n) + s_n\alpha_n\mathbf{x}(n)$$

其中

$$s_n = \begin{cases} +1 & 若分类正确 \\ -1 & 若分类错误 \end{cases}$$

(b)因此，若 $\alpha_n = (1 - s_n\alpha_n)\alpha_{n-1}$ 成立，证明习题开始描述的最优准则满足；这样学习常数 α_n 的最优值为

$$\alpha_n^{\text{opt}} = \frac{\alpha_{n-1}^{\text{opt}}}{1 + s_n\alpha_{n-1}^{\text{opt}}}$$

9.7 第 8 章讨论的最大特征滤波器和自组织特征映射的更新规则都利用 Hebb 学习假设的修正。比较这两个修正，说明它们的不同和相似点。

9.8 良心算法是 SOM 算法的修正，它迫使密度匹配是精确的匹配(DeSieno, 1988)。在表 9-4 小结的良心算法中，每个神经元保存它竞争获胜的次数(即它的突触权值向量在 Euclid 距离下成为距离输入向量最近的神经元的次数)。这里使用的概念，就是如果一个神经元获胜太频繁，它“感到有罪”从而退出竞争。

为了研究利用良心算法在密度匹配上产生的改善，考虑由 20 个神经元组成的一维网格(即线性陈列)利用图 9-19 画出的线性输入密度训练它。

(a)利用计算机仿真比较由良心算法和 SOM 算法产生的密度匹配，对 SOM 算法使用

$\eta=0.005$而良心算法使用 $B=0.0001$，$C=1.0$ 和 $\eta=0.05$。

(b)作为这个比较的参考框架，包括输入密度的“精确”匹配。

480

讨论你的计算机仿真结果。

表 9-4　良心算法小结

1. 寻找和输入向量 $\mathbf{x}$ 最近的突触向量 $\mathbf{w}_i$：

$$\|\mathbf{x}-\mathbf{w}_i\| = \min_j \|\mathbf{x}-\mathbf{w}_j\|, j=1,2,\cdots,N$$

2. 保持一轮神经元竞争获胜的总时间部分 p_j：

$$p_j^{new} = p_j^{old} + B(y_j - p_j^{old})$$

其中 $0<B<<1$ 且

$$y_j = \begin{cases} 1 & 若神经元 j 为获胜神经元 \\ 0 & 其他 \end{cases}$$

在算法开始时，p_j 初始化为零

3. 利用良心机制

$$\|\mathbf{x}-\mathbf{w}_i\| = \min_j (\|\mathbf{x}-\mathbf{w}_j\| - b_j)$$

寻找新的获胜神经元，其中 b_j 是为了修改竞争而引入的偏置项；它定义为

$$b_j = C\left(\frac{1}{N} - p_j\right)$$

其中 C 为偏置因子而 N 为网络中神经元的总数。

4. 更新获胜神经元的突触权值向量：

$$\mathbf{w}_i^{new} = \mathbf{w}_i^{old} + \eta(\mathbf{x}-\mathbf{w}_i^{old})$$

其中 η 为通常在 SOM 算法中使用的学习率参数。

计算机实验

9.9　在这个试验中我们用计算机仿真研究 SOM 算法应用于具有二维输入的一维网格。网格由 65 个神经元组成，输入由图 9-20 所示的三角形内均匀分布的随机点构成。计算由 SOM 算法在 0，20，100，1000，10 000 和 25 000 次迭代后产生的映射。

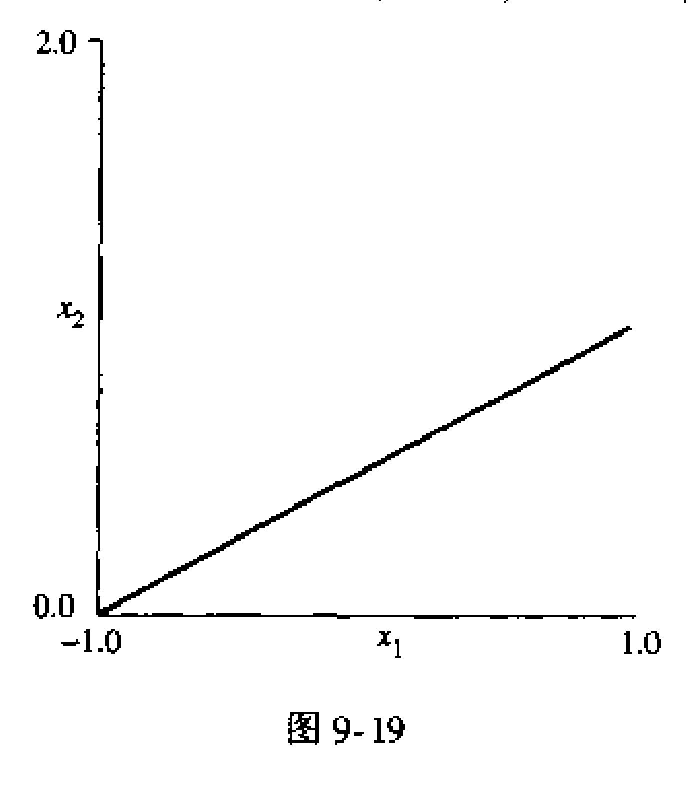


图 9-19

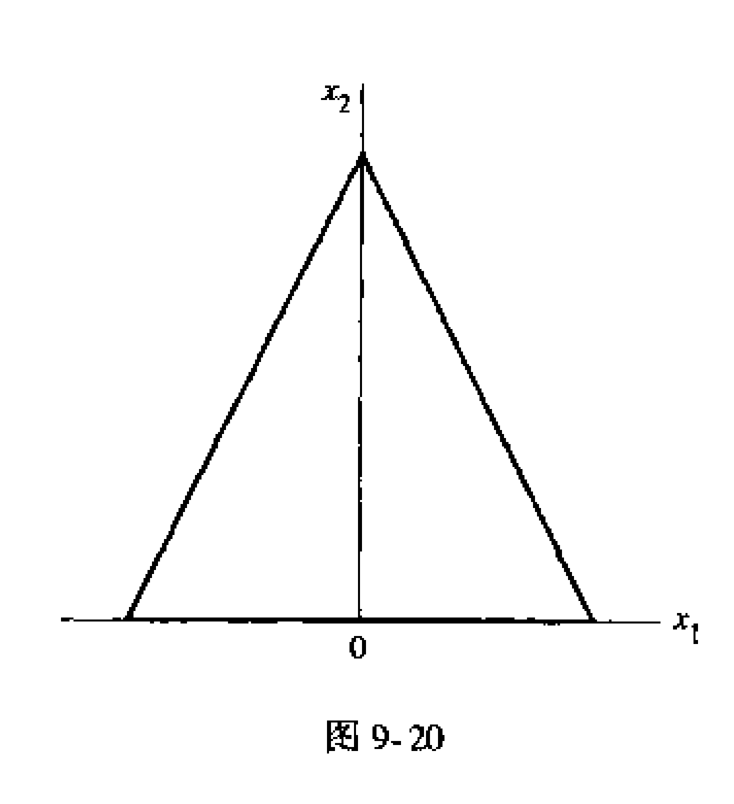


图 9-20

481

9.10　考虑一个用三维输入分布训练的二维神经元网格，网格由 10×10 神经元构成。

(a)在小区域

$$\{(0<x_1<1),(0<x_2<1),(0<x_3<0.2)\}$$

内输入是一致分布的。利用 SOM 算法计算输入空间在 50，1 000 和 10 000 次算法迭代后的二维投影。

(b)当输入在一个更大的区域

$$\{(0 < x_1 < 1),(0 < x_2 < 1),(0 < x_3 < 0.4)\}$$

内均匀分布时重复你的计算。

(c)当输入在立方体

$$\{(0 < x_1 < 1),(0 < x_2 < 1),(0 < x_3 < 1)\}$$

内均匀分布时再一次重复你的计算。

讨论你的计算机仿真结果的含义。

9.11 在 SOM 算法应用中经常出现的问题是不能形成拓扑排序而产生"折叠"映射。当允许邻域体积衰减太快时就会发生这个问题。折叠映射的产生可以看作拓扑排序过程形成某种形式的"局部最小"。

为了研究这个现象，考虑一个 10×20 神经元的二维网格，用在正方形$\{(-1 < x_1 < +1),(-1 < x_2 < +1)\}$内均匀分布的二维输入训练。计算由 SOM 算法产生的映射，允许获胜神经元周围的邻域函数比正常使用的衰减快得多。你可能需要重复几次试验才能看到排序过程的失败。

9.12 SOM 算法的拓扑排序性质可以用于形成高维输入空间的一种抽象的二维表示形式。为了研究这种表示形式，考虑由 10×10 神经元组成的二维网格，它的训练输入空间由 8 维空间的 4 个 Gauss 云$\mathscr{C}_1$，$\mathscr{C}_2$，$\mathscr{C}_3$和$\mathscr{C}_4$构成，它们的中心位置分别为$(0,0,0,\cdots,0)$，$(4,0,0,\cdots,0)$，$(4,4,0,\cdots,0)$和$(0,4,0,\cdots,0)$。计算由 SOM 算法产生的映射，在映射中每个神经元的类别和在该神经元周围输入点中具有最多输入点的类别相同。

9.13 表 9-5 给出重正规化 SOM 算法小结；9.3 节给出算法的简要描述。比较常规的和重正规化的 SOM 算法，注意以下两个问题：

1. 算法实现所涉及的编码复杂性。
2. 训练花费的计算机时间。

表 9-5 重正规化训练算法小结(一维的形式)

1. *初始化*。置码字向量的数目为一小整数(例如，为简单起见使用 2 或对所求问题更具代表性的其他数目)。从训练集中随机选择相应数目的训练向量初始化它们的位置。
2. *选择一个输入向量*。从训练集中随机选择一个输入向量。
3. *输入向量编码*。确定获胜码字向量(即获胜神经元的突触权值向量)。为了做到这一点，在需要时使用"最近邻"或"最小失真"编码规定。
4. *码书更新*。执行通常的"获胜者和它的拓扑邻域"更新。你会发现保持学习率参数η固定(如 0.125)就足够了。例如更新获胜神经元使用η而它的最近邻使用$\eta/2$。
5. *码书分裂*①。继续码书更新(第 4 步)，每次使用随机训练集中挑选的新输入向量直到码书更新的次数是码字向量数目的 10－30 倍。这时码书大概已经稳定，应该进行码书分裂。为做到这一点你既可以采用你所有的码字向量的 Peano 串，且对它们的位置进行插值以产生对 Peano 串的更小粒度的逼近；也可以简单对每两个已有的码字向量连线添加另外码字向量。
6. *训练完成*。继续进行码书更新和码书分裂直到码字向量总数达到某一预定值(如 100)，这时整个训练结束。

① 码书分裂近似在每一回合时加倍码字向量的数目，所以达到任何预定的码字数目无需花费许多的回合。

说明这两种算法的比较，利用从一个正方形内的均匀分布中抽取的数据，且按照下列两个网络配置：

(a)257 个神经元的一维网格

(b)2094 个神经元的一维网格

在这两种情形都以 2 个码字向量开始。

9.14 考虑图 9-21 所示的信号空间图对应的 M 行冲击幅度调制(M-ary pulse-amplitude modulation, PAM)，$M=8.0$。信号点对应于 Gray 编码数据块。每个信号点由具有合适幅度大小的矩形冲击信号表示：

482

$$p(t)=\pm\frac{7}{2},\pm\frac{5}{2},\pm\frac{3}{2},\pm\frac{1}{2},\qquad 0\leqslant t\leqslant T$$

其中 T 为信号区间。在接收器输入，对具有变化信噪比(signal-to-noise ratio, SNR)的传输信号添加零均值的 Gauss 白噪声。SNR 定义为传输信号能量平均和噪声能量平均的比值。

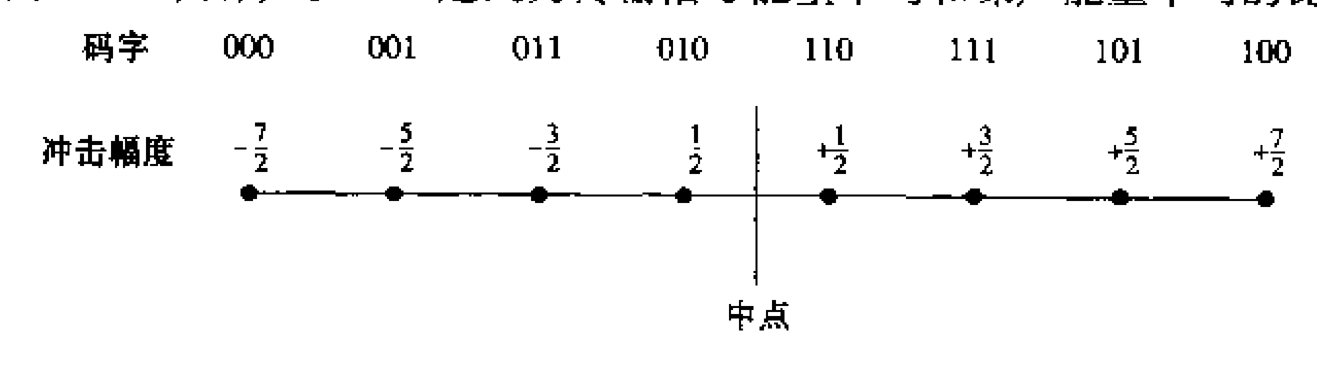


图 9-21

(a)利用随机二值序列作为发送器输入，产生表示SNR = 10,20,30 分贝接收信号的数据。

(b)对这些 SNR，建立自组织特征映射。你可使用的典型值为：

- 对接受信号以 8 倍信号率采样获得的 8 个元素构成输入向量(即每个信号区间 8 个样本)。假设不知道时间信息。
- 64 个神经元的一维网格(即输入向量大小的 8 倍)。

483

(c)对三个 SNR 显示特征映射，由此表示 SOM 算法的拓扑排序性质。

第10章　信息论模型

10.1　简介

Claude Shannon 在 1948 年发表的经典论文中，为信息论打下了基础。Shannon 在信息论方面的开创性工作[1]和其他的研究工作者对它的补充，是对电子工程师设计高效可靠通信系统的需求的直接回应。无论它的实际起源是什么，如我们今天所知道的信息论正是关于通信过程本质的深刻数学理论。这个理论提供一个对根本问题研究的总体框架，例如，信息表示的效率以及一个通信信道可靠信息传输的极限问题。而且该理论包括很多有力的定理用以计算最佳表示和信号所携带信息的传输的理想界限。这些界限非常重要，因为它们为提高信息处理系统的设计提供了标准。

这一章我们的主要目的是讨论以一种原则性方式导致自组织的*信息论模型*。在这个背景下，特别值得注意的模型是由 Linsker 于 1988 年提出的*最大互信息原则*(maximum mutual information principle)[2]。该原则表明，多层神经网络的突触联结以这样一种方式进行：*在网络的每个处理阶段，当进行信号变换时，为保留的信息量达到最大，要遵从一定的约束条件*。利用信息论来解释人们的感知过程并不是什么新的想法[3]。例如，我们可能注意到 1954 年 Attneave 写的一篇早期论文，其中提出了关于感知系统的下面信息理论性作用：

> *感知机制的一个主要功能是减少刺激的冗余，以一种比它冲击接受器的形式更经济的方式对信息进行描述或编码。*

在 Attneave 的论文背后的主要思想在于认识到为减少冗余对场景数据编码和确认场景中特定特征是相关的。这种重要认识和在 Craik(1943)描述的关于人脑的观点相关，在该论文中构造一个外部世界的模型以便结合现实的规则和约束。

484

本章的组织

本章主体组织成两部分。第一部分由 10.2 节至 10.5 节组成，提供对信息论基本原理的回顾。在 10.2 节讨论作为信息的一个定量度量的熵的概念，这自然导致 10.3 节讨论的最大熵原则。其次，我们在 10.4 节讨论互信息的概念和它的性质，随后在 10.5 节讨论 Kullback-Leibler 散度。

本章第二部分由 10.6 节至 10.14 节组成，处理用于自组织系统的信息论模型。10.6 节提出把互信息量作为一个最优化的目标函数。最大互信息原则在 10.7 节介绍，随后讨论该原则与 10.8 节中的冗余减少原则之间的关系。10.9 节与 10.10 节中处理最大互信息原则适应于图像处理中不同应用的两个变体。10.11 节到 10.14 节提出三种不同的方法解决盲源分离问题。

在 10.15 节中提出一些最后的评论结束本章。

10.2 熵

遵循概率论中通常使用的术语，我们以大写字母表示随机变量，以相应的小写字母表示随机变量的值。

对于一个随机变量 X，它的每一个实现(出现)可看作一个消息。严格地说，如果随机变量 X 的幅度值是连续的，则它带有无穷的信息。但是，从物理和生物的角度来看，我们认识到讨论具有无限精度的幅度度量的信息是没有意义的，这就是说可以把 X 的值一致量化到有限的离散水平。这样我们可以把 X 看成是离散的随机变量，其模型为

$$X = \{x_k \mid k = 0, \pm 1, \cdots, \pm K\} \tag{10.1}$$

其中 x_k 是一个离散的数值且$(2K+1)$是总的离散水平。离散水平之间的间隔 δx 假设非常小，能够以足够的精度来描述我们感兴趣的变量。当然我们能够接近连续的极限，只要 $\delta x \to 0$ 且 K 趋于无穷，在这种情况下就得到连续变量而且(在本节后面部分我们将看到)求和变成积分。

为完善模型，让事件 $X = x_k$ 以概率

485

$$p_k = P(X = x_k) \tag{10.2}$$

发生，其中要求

$$0 \leqslant p_k \leqslant 1 \text{ 和 } \sum_{k=-K}^{K} p_k = 1 \tag{10.3}$$

假如事件 $X = x_k$ 发生的概率 $p_k = 1$，因此要求对所有 $i \neq k$ 有 $p_i = 0$。在这种情况下，如果事件 $X = x_k$ 发生就没有什么“惊奇”的了，并且不传达任何“信息”，因为我们知道消息必须是什么。在另一种情况下，如果各种离散水平发生的概率不同，特别地概率 p_k 很小，那么当 X 取值 x_k 而不是具有更高概率 p_i 的离散水平 $x_i(i \neq k)$时，这就有更大的“惊奇”和有“信息”了。因此词“不确定”、“惊奇”和“信息”是相关的。在 $X = x_k$ 发生之前，有一定的不确定性。在 $X = x_k$ 发生之后，有一定惊奇。在 $X = x_k$ 发生之后，信息量增加了。这里的三个量很显然是一样的，而且信息量与事件发生的概率成反比。

我们定义观察到具有概率 p_k 的事件 $X = x_k$ 后所获得的信息增益量为对数函数

$$I(x_k) = \log\left(\frac{1}{p_k}\right) = -\log p_k \tag{10.4}$$

其中对数函数的底是任意的。当以自然对数为底时，信息的单位是奈特(nat)，当以 2 为底时，单位是比特(bit)。在任何情况下以式(10.4)定义的信息量都有以下的性质：

1. $$I(x_k) = 0, \text{ 当 } p_k = 1 \tag{10.5}$$

显然，如果我们绝对肯定将发生的事件，则当其发生时就没有获得信息。

2. $$I(x_k) \geqslant 0, \text{ 当 } 0 \leqslant p_k \leqslant 1 \tag{10.6}$$

也就是说，当事件 $X = x_k$ 发生时，或提供一些信息或不提供信息，但不会导致信息损失。

3. $$I(x_k) > I(x_i), \text{ 当 } p_k < p_i \tag{10.7}$$

也就是说，小概率事件发生时携带的信息量比大概率事件发生时携带的信息量多。

信息量 $I(x_k)$也是一个具有概率 p_k 的离散随机变量。$I(x_k)$在全部 $2K+1$ 个离散数值上

的平均值定义为

$$H(X) = E[I(x_k)] = \sum_{k=-K}^{K} p_k I(x_k) = -\sum_{k=-K}^{K} p_k \log p_k \tag{10.8}$$

486

量 $H(X)$叫做一个可取有限离散值的随机变量 X 的熵；之所以称为熵是因为(10.8)给出的定义与统计热力学中的熵非常相似[4]。熵 $H(X)$表示每一个消息所携带的信息的平均量。注意在 $H(X)$中 X 不是$H(X)$的变量，而是一个随机变量的标记。同时注意到在式(10.8)中我们取 0 log 0 为 0。

熵 $H(X)$被限定如下：

$$0 \leq H(X) \leq \log(2K+1) \tag{10.9}$$

其中$(2K+1)$是总的离散水平的数目。进一步，我们作如下说明：

1. $H(X)=0$ 当且仅当对于某一个 k 概率$p_k=1$ 时，而集合中其他的概率为 0；熵的这个下界不对应不确定性。

2. $H(X)=\log_2(2K+1)$当且仅当对所有的 k，$p_k=1/(2K+1)$(即所有的离散值的概率相等)；这个上界对应最大不确定性。

第二性质的证明要用到下面的引理(Gray,1990)：

对离散的随机变量 X 给定任意两个分布$\{p_k\}$和$\{q_k\}$，则

$$\sum_k p_k \log\left(\frac{p_k}{q_k}\right) \geq 0 \tag{10.10}$$

当且仅当对所有的 k，$p_k=q_k$ 都成立时，上面的等式成立。

这个引理所用的量是如此的重要，以致我们停下来以适宜在随机系统的研究中使用的形式描述它。令 $p_X(x)$和 $q_X(x)$表示一个随机变量 X 在两个操作条件下处于状态x 的概率。两个概率质量函数 $p_X(x)$和 $q_X(x)$的相对熵或 Kullback-Leibler 散度(距离)定义如下(Kullback, 1968, Gray, 1990; Cover and Thomas, 1991)：

$$D_{p\|q} = \sum_{x\in\mathscr{X}} p_X(x)\log\left(\frac{p_X(x)}{q_X(x)}\right) \tag{10.11}$$

其中求和是对所有的可能的系统状态(即离散随机变量 X 的字母表$\mathscr{X}$)。概率质量函数 $q_X(x)$起着参考度量的作用。

487

连续随机变量的微分熵

信息论概念的讨论现在只涉及它们的幅度离散的随机变量总体。现在我们将这些概念中的一些扩展到连续随机变量。

假设连续随机变量 X 的概率密度函数是$f_X(x)$，与离散随机变量的熵的定义类似，我们作如下定义：

$$h(X) = -\int_{-\infty}^{\infty} f_X(x)\log f_X(x)\,dx = -E[\log f_X(x)] \tag{10.12}$$

我们将 $h(X)$定义为 X 的微分熵(differential entropy)，与一般的或绝对熵相区别。我们这样做认识到虽然 $h(X)$是一个有用的数学量，但它在任何意义下也不是一种 X 的随机性度量。

我们对使用(10.12)的合理性可以解释如下。开始将连续随机变量 X 看成离散随机变量

的极限形式，设 $x_k = k\delta x$，其中 $k=0, \pm 1, \pm 2, \cdots$，且 δx 趋于0。由定义，连续随机变量 X 取值在$[x_k, x_k+\delta x]$之间的概率为 $f_X(x_k)\delta x$。所以，当 δx 趋于0时连续随机变量 X 的普通熵可以写成如下极限的形式：

$$
\begin{aligned}
H(X) &= -\lim_{\delta x \to 0} \sum_{k=-\infty}^{\infty} f_X(x_k)\delta x \log(f_X(x_k)\delta x) \\
&= -\lim_{\delta x \to 0}\left[\sum_{k=-\infty}^{\infty} f_X(x_k)(\log f_X(x_k))\delta x + \log\delta x \sum_{k=-\infty}^{\infty} f_X(x_k)\delta x\right] \\
&= -\int_{-\infty}^{\infty} f_X(x)\log f_X(x)\,dx - \lim_{\delta x \to 0}\log\delta x \int_{-\infty}^{\infty} f_X(x)\,dx \\
&= h(X) - \lim_{\delta x \to 0}\log\delta x
\end{aligned}
\tag{10.13}
$$

其中最后一行用到了式(10.12)以及在概率密度函数下方的总面积为1这个事实。当 δx 趋于0时，$-\log\delta x$ 趋于无穷大。这意味着连续随机变量的熵是无穷大。在直觉上，我们也期望这是真的，因为随机变量可以在$(-\infty, \infty)$上任意取值，和随机变量相关联的不确定性是无穷阶的。为了避免出现项 $-\log\delta x$ 所带来的问题，我们采用 $h(X)$作为描述随机变量 X 的微分熵，项 $-\log\delta x$ 作为参考。而且，由于熵作为一个随机系统处理的信息实体，我们感兴趣的实际上是具有相同参考的两个熵项的差，信息将和相应微分熵项之间的差是一样的。所以我们完全有理由采用在(10.13)所定义的项 $h(X)$作为连续随机变量 X 的微分熵。

当有一个由 n 个随机变量 $X_1, X_2, \cdots, X_n$ 组成的随机连续向量 $\mathbf{X}$ 时，我们定义 $\mathbf{X}$ 的微分熵为 n 重积分

488

$$
h(\mathbf{X}) = -\int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\log f_{\mathbf{X}}(\mathbf{x})\,d\mathbf{x} = -E[\log f_{\mathbf{X}}(\mathbf{x})] \tag{10.14}
$$

其中 $f_{\mathbf{X}}(\mathbf{x})$是 $\mathbf{X}$ 的联合概率密度函数。

例 10.1 均匀分布 考虑在[0,1]区间上均匀分布的随机变量 X，表示为

$$
f_X(x) = \begin{cases} 1 & 0 \leq x \leq 1 \\ 0 & \text{其他} \end{cases}
$$

应用(10.12)，我们得到 X 的微分熵为

$$
h(X) = -\int_{-\infty}^{\infty} 1 \cdot \log 1\,dx = -\int_{-\infty}^{\infty} 1 \cdot 0\,dx = 0
$$

所以 X 的熵为0。 ■

微分熵的性质

从式(10.12)给出的微分熵 $h(X)$的定义中容易看出平移不会改变它的值，即

$$
h(X+c) = h(X) \tag{10.15}
$$

其中 c 为常量。

$h(X)$另一个有用的性质是

$$
h(aX) = h(X) + \log|a| \tag{10.16}
$$

其中 a 为比例系数。为了证明该式，我们首先知道概率密度函数曲线下方的面积是1，故

$$
f_Y(y) = \frac{1}{|a|}f_Y\left(\frac{y}{a}\right) \tag{10.17}
$$

接着应用式(10.12)，我们可写成

$$h(Y) = -E[\log f_Y(y)] = -E\left[\log\left(\frac{1}{|a|}f_Y\left(\frac{y}{a}\right)\right)\right] = -E\left[\log f_Y\left(\frac{y}{a}\right)\right] + \log|a|$$

代入 $Y = aX$ 得到

489

$$h(aX) = -\int_{-\infty}^{\infty} f_X(x)\log f_X(x)\,dx + \log|a|$$

由此立刻得出式(10.16)。

式(10.16)用于纯量的随机变量，也可以推广用于随机向量 $\mathbf{X}$ 乘以矩阵 $\mathbf{A}$ 的情况如下：

$$h(\mathbf{AX}) = h(\mathbf{X}) + \log|\det(\mathbf{A})| \tag{10.18}$$

其中 $\det(\mathbf{A})$ 是矩阵 $\mathbf{A}$ 的行列式。

10.3 最大熵原则

假设有一个随机系统，已知一组状态，但不知其概率，而且我们知道这些状态的概率分布的一些限制条件。这些条件或者是已知一定的总体平均值，或者是它们的一些界限。在给定关于模型的先验知识的条件下，问题是选择一个在某种意义下最佳的概率模型。我们经常发现有无穷多种模型可以满足条件。应该选择哪个模型呢？

这个基本问题的答案基于 Jaynes(1957)提出的*最大熵原则*[5]。最大熵原则可以陈述如下(Jaynes,1957,1982)：

当根据不完整的信息作为依据进行推断时，应该由满足分布限制条件的具有最大熵的概率分布推得。

实际上，熵的概念在概率分布空间定义一种度量，使得具有较高熵的分布比其他的分布具有更大的值。

从上面陈述，很明显"最大熵问题"是一个约束最优化问题。为了说明解这个问题的步骤，考虑最大微分熵

$$h(X) = -\int_{-\infty}^{\infty} f_X(x)\log f_X(x)\,dx$$

对所有随机变量 X 的概率密度函数 $f_X(x)$，并满足以下约束条件：

1. $f_X(x) \geqslant 0$，在 x 的支撑集之外等式成立

2. $\displaystyle\int_{-\infty}^{\infty} f_X(x)\,dx = 1$

3. $\displaystyle\int_{-\infty}^{\infty} f_X(x)g_i(x)\,dx = \alpha_i \qquad$ 对于 $i = 1,2,\cdots,m$

其中 $g_i(x)$ 是 x 的一个函数。约束 1 和约束 2 描述概率密度函数的基本属性，约束 3 定义变量 X 的矩，它随函数 $g_i(x)$ 的表达式不同而发生变化。实际上，约束 3 综合随机变量 X 的可用先验知识。为了解决这个约束最优化问题，我们利用 Lagrange 乘子法[6]，首先形成目标函数

490

$$J(f) = \int_{-\infty}^{\infty}\left[-f_X(x)\log f_X(x) + \lambda_0 f_X(x) + \sum_{i=1}^{m}\lambda_i g_i(x)f_X(x)\right]dx \tag{10.19}$$

其中$\lambda_0,\lambda_1,\cdots,\lambda_m$是 Lagrange 乘子。对被积函数求$f_X(x)$的微分，并使其为0，我们得到

$$-1-\log f_X(x)+\lambda_0+\sum_{i=1}^{m}\lambda_i g_i(x)=0$$

解此方程得

491

$$f_X(x)=\exp\left(-1+\lambda_0+\sum_{i=1}^{m}\lambda_i g_i(x)\right) \tag{10.20}$$

在式(10.20)的 Lagrange 乘子根据约束条件2和3选择。式(10.20)定义这个问题的最大熵分布。

例 10.2 一维 Guass 分布 假设我们可用的先验知识为随机变量 X 的均值 μ 和方差 σ^2。根据定义，得到

$$\int_{-\infty}^{\infty}(x-\mu)^2 f_X(x)dx=\sigma^2=\text{constant}$$

将此式与约束条件3作比较，看出

$$g_1(x)=(x-\mu)^2,\quad \alpha_1=\sigma^2$$

所以应用(10.20)可得

$$f_X(x)=\exp[-1+\lambda_0+\lambda_1(x-\mu)^2]$$

注意如果$f_X(x)$和$(x-\sigma)^2f_X(x)$对 x 的积分是收敛的，则 λ_1 为负数。将此等式代入约束条件2和3，解出 λ_0 和 λ_1 我们得到

$$\lambda_0=1-\log(2\pi\sigma^2),\quad \lambda_1=-\frac{1}{2\sigma^2}$$

所以希望的$f_X(x)$的分布形式为

$$f_X(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{10.21}$$

我们知道这是一个均值为 μ 和方差为 σ^2 的 Gauss 随机变量 X 的概率密度函数。这样的随机变量的微分熵的最大值为

$$h(X)=\frac{1}{2}[1+\log(2\pi\sigma^2)] \tag{10.22}$$

对这个例子我们作如下的小结：

1. 对于给定的方差 σ^2，在任意的随机变量中 Gauss 随机变量取得微分熵的最大值。也就是说，如果 X 是一个 Gauss 随机变量，Y 是其他具有相同均值和方差的随机变量，则对所有的 Y

$$h(X)\geqslant h(Y)$$

只有当 X 与 Y 相同时等式成立。

2. Gauss 随机变量 X 的熵值取决于 X 的方差(即与 X 的均值无关)。

例 10.3 多维 Gauss 分布 在这第二个例子中，我们想在例10.2的结果基础上，建立计算多维 Gauss 分布的微分熵的计算公式。由于 Gauss 分布的熵与随机变量 X 的均值无关，为简化讨论，我们可以仅讨论具有均值为0的随机变量 $\mathbf{X}$。这样 $\mathbf{X}$ 的二阶统计性质由其协方差矩阵 $\mathbf{\Sigma}$ 决定，它为 $\mathbf{X}$ 同自身的外积的期望。这样 $\mathbf{X}$ 的联合概率密度函数由

$$f_{\mathbf{X}}(\mathbf{x})=\frac{1}{(2\pi)^{m/2}(\det(\mathbf{\Sigma}))^{1/2}}\exp\left(-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}\right) \tag{10.23}$$

给出(Wilks,1962)，其中 $\det(\mathbf{\Sigma})$ 是 $\mathbf{\Sigma}$ 的行列式。式(10.14)定义 $\mathbf{X}$ 的微分熵。因此将(10.23)代入(10.14)，我们得到

$$h(\mathbf{X}) = \frac{1}{2}[m + m\log(2\pi) + \log|\det(\mathbf{\Sigma})|] \tag{10.24}$$

这包括式(10.22)作为其特例。按最大熵原则的观点，我们可以这样说，对于给定的一个协方差矩阵 $\mathbf{\Sigma}$，在所有零均值随机向量可达到的微分熵中，多元 Gauss 分布具有最大的微分熵，此最大微分熵由式(10.24)定义。 ■

10.4 互信息

在设计一个自组织系统时，根本的目的就是仅仅根据输入模式来获得一个学习算法，该算法能够学习输入和输出的关系。在这个背景下，由于互信息的概念有很多好的性质，所以非常重要。为了以后的讨论，假定随机系统具有输入 X 和输出 Y，而且 X 和 Y 只允许取离散的值，分别由 x 和 y 表示。熵 $H(X)$ 表示 X 的先验不确定性。那么，当观测到 Y 后我们如何度量对 X 的不确定性？为了回答此问题，我们定义在给定 Y 时 X 的条件熵为(Gray, 1990; Cover & Thomas, 1991)

$$H(X|Y) = H(X,Y) - H(Y) \tag{10.25}$$

492

具有性质

$$0 \leqslant H(X|Y) \leqslant H(X) \tag{10.26}$$

条件熵 $H(X|Y)$ 表示在观测到系统输出 Y 后，对 X 保留的不确定性度量。在式(10.25)中 $H(X,Y)$ 是 X 和 Y 的联合熵，由

$$H(X,Y) = -\sum_{x\in\mathscr{X}}\sum_{y\in\mathscr{Y}} p(x,y)\log p(x,y)$$

定义，其中 $p(x,y)$ 是离散随机变量 X 和 Y 的联合概率质量函数，而 $\mathscr{X}$ 和 $\mathscr{Y}$ 表示它们各自的字母表。

由于熵 $H(X)$ 表示在没有观测系统输出前我们对系统输入的不确定性，条件熵 $H(X|Y)$ 表示在观测到系统输出后对系统输入的不确定性，差 $H(X) - H(X|Y)$ 表示观察到系统输出之后我们对系统输入的不确定性的减少。这个量就叫做随机变量 X 和 Y 之间的互信息，由 $I(X;Y)$ 表示，我们可以写成[7]

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= \sum_{x\in\mathscr{X}}\sum_{y\in\mathscr{Y}} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \end{aligned} \tag{10.27}$$

熵是互信息的一个特例，因为我们有

$$H(X) = I(X;X)$$

两个离散随机变量 X 和 Y 的互信息 $I(X;Y)$ 有如下的性质(Cover and Thomas, 1991; Gray, 1990)：

1. X 和 Y 的互信息具有对称性；也即

$$I(Y;X) = I(X;Y)$$

其中互信息 $I(Y;X)$ 表示观察系统输入 X，对系统输出 Y 的不确定性的减少，而 $I(X;Y)$ 表示观测系统输出后对系统输入的不确定性的减少。

2. X 和 Y 的互信息总是非负的；也即

$$I(X;Y) \geq 0$$

实际上，这个性质说明，通过观测系统的输出 Y，平均说来我们不可能丢失信息。而且，当且仅当输入和输出统计独立时互信息为0。

493

3. X 和 Y 的互信息也可以用 Y 的熵表示为

$$I(X;Y) = H(Y) - H(Y \mid X) \tag{10.28}$$

其中 $H(Y|X)$ 是条件熵。式(10.28)的右端表示系统输出 Y 的总体平均传达信息减去我们知道系统输入 X 后关于 Y 的总体平均传达信息 $I(X;Y)$。后一个量 $H(Y|X)$ 传达关于处理噪声而不是关于系统输入 X 的信息。

图10-1用一个可视化的图来解释等式(10.27)和(10.28)。系统的输入 X 的熵 $H(X)$ 用左边的圆表示，输出 Y 的熵 $H(Y)$ 用右边的圆表示，X 和 Y 的互信息用图中的两圆的交集表示。

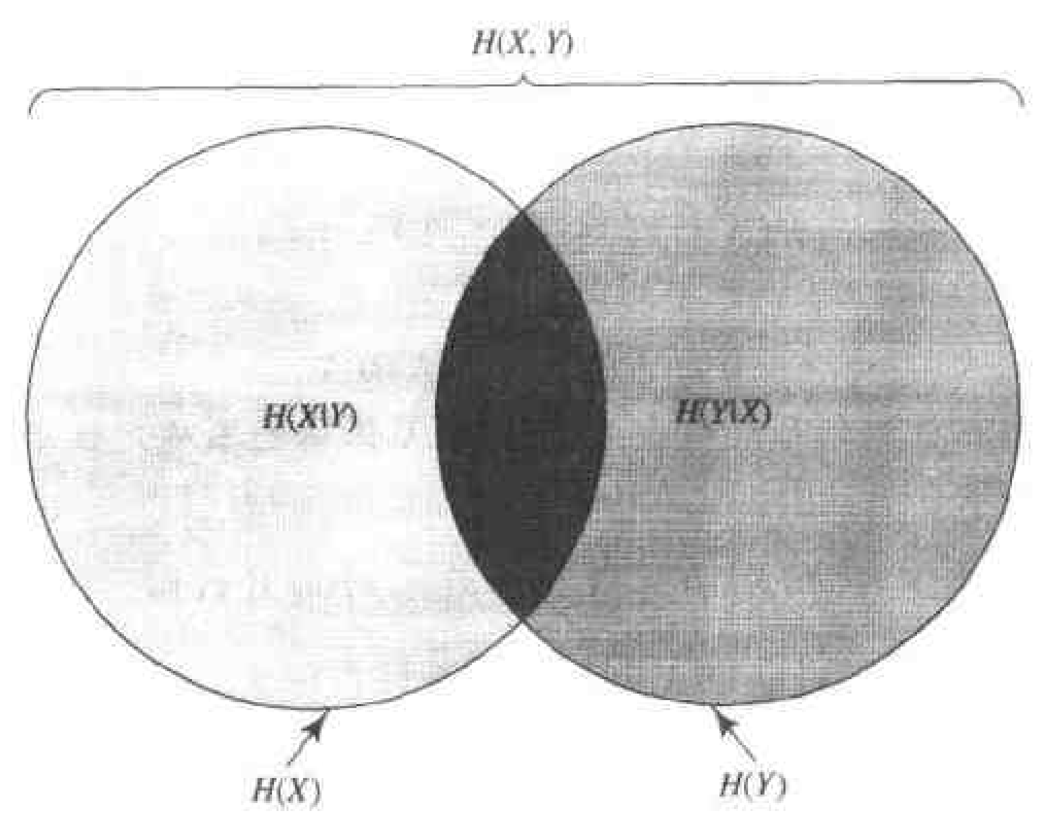


图10-1 互信息 $I(X;Y)$ 和熵 $H(X)$ 及熵 $H(Y)$ 的关系说明

连续随机变量的互信息

给定一对连续的随机变量 X 和 Y，类似式(10.27)，我们定义随机变量 X 和 Y 的互信息为

$$I(X;Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{X,Y}(x,y)\log\left(\frac{f_X(x \mid y)}{f_X(x)}\right)dx\,dy \tag{10.29}$$

其中 $f_{X,Y}(x,y)$ 是 X 和 Y 联合概率密度函数，$f_X(x|y)$ 是当 $Y=y$ 时 X 的条件概率密度函数。注意

$$f_{X,Y}(x,y) = f_X(x \mid y)f_Y(y)$$

所以我们也可以写成

$$I(X;Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{X,Y}(x,y)\log\left(\frac{f_{X,Y}(x,y)}{f_X(x)f_Y(y)}\right)dx\,dy$$

同前面讨论的离散随机变量类似，连续随机变量 X 和 Y 的互信息有如下的性质：

$$
\begin{aligned}
I(X;Y) &= h(X) - h(X \mid Y) \\
&= h(Y) - h(Y \mid X) \\
&= h(X) + h(Y) - h(X,Y)
\end{aligned} \tag{10.30}
$$

494

$$I(Y;X) = I(X;Y) \tag{10.31}$$

$$I(X;Y) \geqslant 0 \tag{10.32}$$

参量 $h(X)$是 X 的微分熵，同 $h(Y)$一样。参量 $h(X|Y)$是给定 Y 时 X 的条件微分熵，由重积分

$$h(X \mid Y) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{X,Y}(x,y)\log f_X(x \mid y)\, dx\, dy \tag{10.33}$$

定义。参量 $h(Y|X)$是给定 X 时 Y 的条件微分熵，定义与 $h(X|Y)$类似。参量 $h(X,Y)$是 X 和 Y 的联合微分熵。

注意式(10.32)，只有在随机变量 X 和 Y 统计独立时等式才成立。当满足此条件时，X 和 Y 的联合概率密度函数可分解成

$$f_{X,Y}(x,y) = f_X(x)f_Y(y) \tag{10.34}$$

其中 $f_X(x)$和 $f_Y(y)$分别是 X 和 Y 的边沿概率密度函数。等价地，我们写成

$$f_X(x \mid y) = f_X(x)$$

这就是说 Y 的结果的知识完全不能影响 X 的分布。将其代入式(10.29)导致 $I(X;Y)=0$。

在式(10.29)中给出的互信息适用于纯量随机变量 X 和 Y。这个定义也易于扩展至随机向量 $\mathbf{X}$ 和 $\mathbf{Y}$，因此我们可以写成 $I(\mathbf{X};\mathbf{Y})$。特别地，我们定义 $I(\mathbf{X};\mathbf{Y})$为多重积分：

$$I(\mathbf{X};\mathbf{Y}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})\log\left(\frac{f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{y})}{f_{\mathbf{X}}(\mathbf{x})}\right) d\mathbf{x}\, d\mathbf{y} \tag{10.35}$$

互信息 $I(\mathbf{X};\mathbf{Y})$同样具有与式(10.30)至式(10.32)的关于纯量随机变量性质平行的性质。

10.5 Kullback-Leibler 散度

在式(10.11)中我们定义离散随机变量 Kullback-Leibler 散度。这个定义也可扩展到随机向量的一般情况。$f_{\mathbf{X}}(\mathbf{x})$和 $g_{\mathbf{X}}(\mathbf{x})$表示 $m \times 1$ 的随机向量 $\mathbf{X}$ 的两个不同的概率分布函数，根据式(10.11)，我们可以定义$f_{\mathbf{X}}(\mathbf{x})$和 $g_{\mathbf{X}}(\mathbf{x})$的 Kullback-Leibler 散度为(Kullback,1968;Shore and Johnson,1980)

495

$$D_{f_{\mathbf{X}} \| g_{\mathbf{X}}} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\log\left(\frac{f_{\mathbf{X}}(\mathbf{x})}{g_{\mathbf{X}}(\mathbf{x})}\right) d\mathbf{x} \tag{10.36}$$

Kullback-Leibler 散度有一些特有的性质：

1. 它总是正的或为零。在特殊的条件下，当 $f_{\mathbf{X}}(\mathbf{x}) = g_{\mathbf{X}}(\mathbf{x})$时，两个分布完全重合，而 $D_{f \| g}$正好为零。

2. 对于向量 $\mathbf{x}$ 的各分量作如下的改变，其值不变：

- 各分量依序置换
- 乘以一个比例系数
- 单调非线性变换

一对向量 $\mathbf{X}$，$\mathbf{Y}$ 之间的互信息 $I(\mathbf{X};\mathbf{Y})$用 Kullback-Leibler 散度有一个有趣的解释。首先，

我们注意到

$$f_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y}) = f_{\mathbf{Y}}(\mathbf{y} \mid \mathbf{x}) f_{\mathbf{X}}(\mathbf{x}) \tag{10.37}$$

所以，可以将式(10.35)改写成如下的等价形式：

$$I(\mathbf{X};\mathbf{Y}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y}) \log\left(\frac{f_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})}{f_{\mathbf{X}}(\mathbf{x}) f_{\mathbf{Y}}(\mathbf{y})}\right) d\mathbf{x}\, d\mathbf{y}$$

将其与式(10.36)作比较。我们立即推得

$$I(\mathbf{X};\mathbf{Y}) = D_{f_{\mathbf{X},\mathbf{Y}} \| f_{\mathbf{X}} f_{\mathbf{Y}}} \tag{10.38}$$

总的来说，$\mathbf{X}$ 和 $\mathbf{Y}$ 之间的互信息等于联合概率密度函数 $f_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})$ 以及概率密度函数 $f_{\mathbf{X}}(\mathbf{x})$ 和 $f_{\mathbf{Y}}(\mathbf{y})$ 的乘积的 Kullback-Leibler 散度。

后一结果的特例是 $m \times 1$ 的随机向量 $\mathbf{X}$ 的概率密度函数 $f_{\mathbf{X}}(\mathbf{x})$ 和它的 m 个边缘概率密度函数的 Kullback-Leibler 散度。令 $\tilde{f}_{X_i}(x_i)$ 表示第 i 个元素 X_i 的边缘概率密度函数，由

$$\tilde{f}_{X_i}(x_i) = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x}^{(i)}, \qquad i = 1,2,\cdots,m \tag{10.39}$$

定义，其中 $\mathbf{x}^{(i)}$ 是一个从 $\mathbf{x}$ 中除去第 i 个元素后的 $(m-1) \times 1$ 向量。$f_{\mathbf{X}}(\mathbf{x})$ 和析因分布 $\prod_i \tilde{f}_{X_i}(x_i)$ 的 Kullback-Leibler 散度定义为

$$D_{f_{\mathbf{X}} \| \tilde{f}_{\mathbf{X}}} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{f_{\mathbf{X}}(\mathbf{x})}{\prod_{i=1}^{m} \tilde{f}_{X_i}(x_i)}\right) d\mathbf{x} \tag{10.40}$$

也可以写成展开形式

496

$$D_{f_{\mathbf{X}} \| \tilde{f}_{\mathbf{X}}} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log f_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x} - \sum_{i=1}^{m} \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log \tilde{f}_{X_i}(x_i)\, d\mathbf{x} \tag{10.41}$$

按定义，式(10.41)右边第一个积分等于 $-h(\mathbf{X})$，其中 $h(\mathbf{X})$ 是 $\mathbf{X}$ 的微分熵。为了处理第二项，我们首先注意到

$$d\mathbf{x} = d\mathbf{x}^{(i)} dx_i$$

因此可以写成

$$\int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log \tilde{f}_{X_i}(x_i)\, d\mathbf{x} = \int_{-\infty}^{\infty} \log \tilde{f}_{X_i}(x_i) \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x}^{(i)}\, dx_i \tag{10.42}$$

其中右端内层积分是对 $(m-1) \times 1$ 向量 $\mathbf{x}^{(i)}$ 积分，而外层积分是对标量 x_i 积分。但从(10.39)，我们发现内层积分实际上等于边缘概率密度函数 $\tilde{f}_{X_i}(x_i)$。由此可以将(10.42)重写为等价形式

$$\begin{aligned}\int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log \tilde{f}_{X_i}(x_i)\, d\mathbf{x} &= \int_{-\infty}^{\infty} \tilde{f}_{X_i}(x_i) \log \tilde{f}_{X_i}(x_i)\, dx_i \\ &= -\tilde{h}(X_i), \qquad i = 1,2,\cdots,m\end{aligned} \tag{10.43}$$

其中 $\tilde{h}(X_i)$ 是第 i 个边缘熵(即边缘概率密度函数 $\tilde{f}_{X_i}(x_i)$ 的微分熵)。最后将式(10.43)代入式(10.41)，并注意式(10.41)中的第一个积分为 $-h(\mathbf{X})$，我们将式(10.41)的 Kullback-Leibler 散度化简为

$$D_{f_{\mathbf{X}} \| \tilde{f}_{\mathbf{X}}} = -h(\mathbf{X}) + \sum_{i=1}^{m} \tilde{h}(X_i) \tag{10.44}$$

这个公式将在本章后面讨论盲源分离问题中特别有用。

Pythagoras 分解

下面我们考虑概率密度函数 $f_{\mathbf{X}}(\mathbf{x})$ 和 $f_{\mathbf{U}}(\mathbf{x})$ 之间的 Kullback-Leibler 散度。$m\times 1$ 随机向量 $\mathbf{U}$ 是由 m 个独立的变量组成，由

$$f_{\mathbf{U}}(\mathbf{x}) = \prod_{i=1}^{m} f_{U_i}(x_i)$$

表示，而 $m\times 1$ 的随机变量 $\mathbf{X}$ 通过 $\mathbf{U}$ 定义为

$$\mathbf{X} = \mathbf{AU}$$

其中 $\mathbf{A}$ 是一个非对角矩阵。令 $\tilde{f}_{X_i}(x_i)$ 表示从 $f_{\mathbf{X}}(\mathbf{x})$ 导出的每一个 X_i 的边缘概率密度，则 $f_{\mathbf{X}}(\mathbf{x})$ 和 $f_{\mathbf{U}}(\mathbf{x})$ 之间的 Kullback-Leibler 散度可以作如下的 Pythagoreas 分解：

$$D_{f_{\mathbf{X}}\| f_{\mathbf{U}}} = D_{f_{\mathbf{X}}\| \tilde{f}_{\mathbf{X}}} + D_{\tilde{f}_{\mathbf{X}}\| f_{\mathbf{U}}} \tag{10.45}$$

497

我们之所以称这个经典的关系为 Pythagoreas 分解，是因为它具有信息-几何解释(Amari, 1985)。在注释[8]中给出这种分解的证明。

10.6 互信息作为最优化的目标函数

现在我们对 Shannon 的信息论模型已经有了适当的了解，可以讨论它在研究自组织系统中的作用。

为了进行讨论，设有一个多输入/多输出的神经网络系统。在这里主要目标是为一个特定任务(例如，建模、抽取统计突出特征或信号分离)而设计的系统进行自组织。通过选择某些系统变量间的互信息作为优化的*目标函数*，这个要求可以满足。这种特定的选择应该考虑下述因素：

- 互信息如同 10.4 节的讨论有一些独特的性质。
- 无需教师也可确定，这样自组织的假定自然满足。

问题变成了系统调整自由参数(即突触权值)以优化互信息的问题。

根据应用的不同，我们能够确定如图 10-2 所示的 4 种不同情况，它们都可能在实际中出现。这些情况可以描述如下：

498

- 在 10-2a 描绘的情况 1，输入向量 $\mathbf{X}$ 由分量 $X_1, X_2, \cdots, X_m$ 组成，输出向量 $\mathbf{Y}$ 由分量 $Y_1, Y_2, \cdots, Y_l$ 组成。需求是*最大化传送到系统输出 $\mathbf{Y}$ 的关于系统输入 $\mathbf{X}$ 的信息*。
- 在 10-2b 描绘的情况 2，一对输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 是从相邻但不重叠的图像区域截取而来。各自产生的纯量输出分别是 Y_a 和 Y_b。需求是*最大化传送到 Y_a 的关于 Y_b 的信息*，以及相反的需求。
- 在图 10-2c 描绘的情况 3，输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 是从两幅不同的图像相应部分截取而来。各自产生的输出分别是 Y_a 和 Y_b，需求是*最小化传送到 Y_a 的关于 Y_b 的信息*。
- 在图 10-2d 描绘的情况 4，输入向量 $\mathbf{X}$ 和输出向量 $\mathbf{Y}$ 与图 10-2a 定义的形式相似，但有相同的维数(即 $l=m$)。这里的目标是*使输出向量 $\mathbf{Y}$ 的各分量之间的统计依赖最小化*。

在所有的这些情况下，互信息扮演中心的角色。但是，它的推导过程还是要根据所考虑

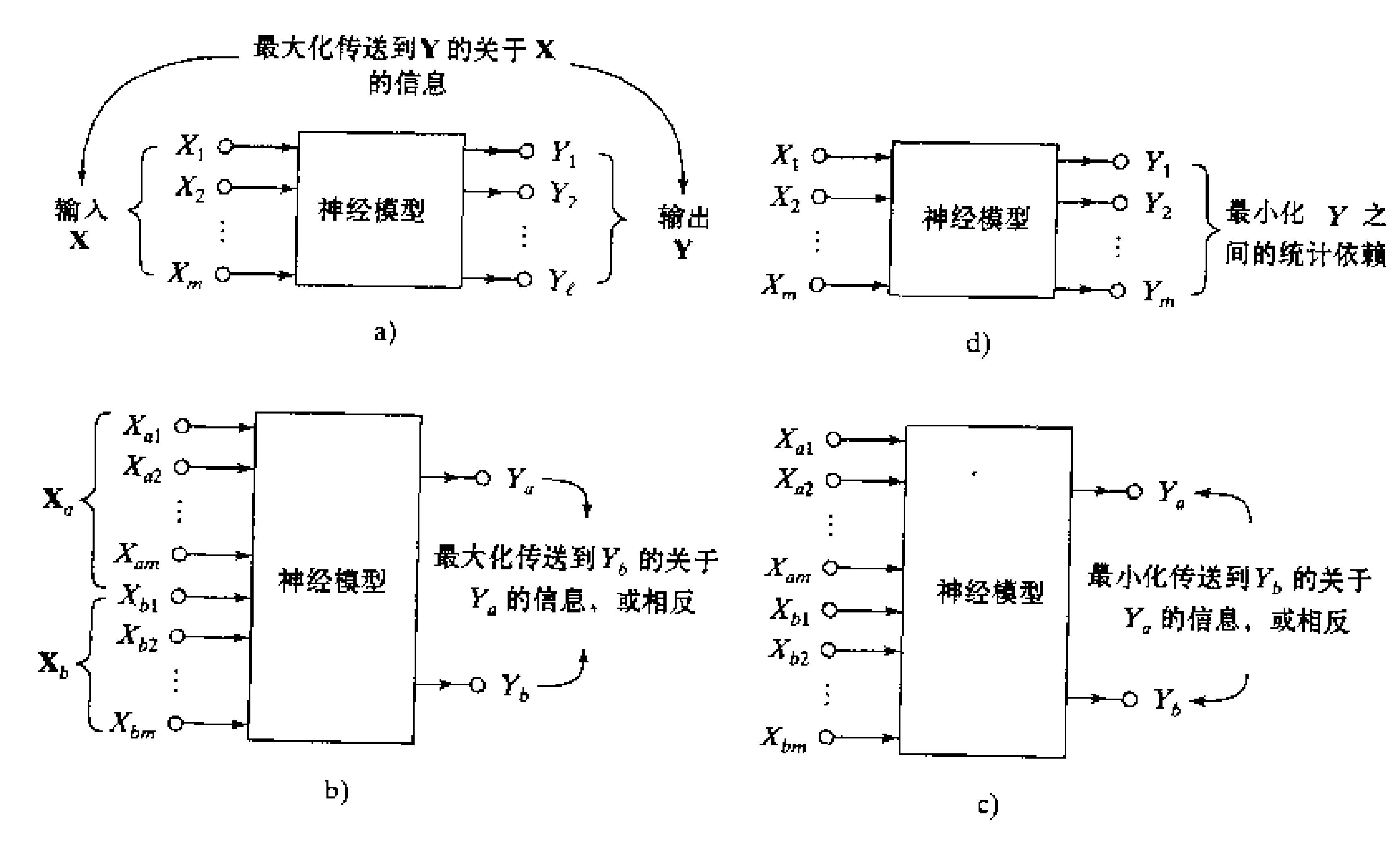


图 10-2 适用于 Infomax 应用及它的三个变体的四个基本情况

的具体情况而定。在本章余下的部分将以刚才罗列的顺序讨论涉及这些情况的问题以及它们的实际含义。

10.7 最大互信息原则

设计一个神经处理器使互信息 $I(\mathbf{Y};\mathbf{X})$最大的思想作为统计信号处理的基础是吸引人的。这种优化方法在 Linsker(1987,1988a,1989a)提出的*最大互信息*(maximum mutual information (Infomax))原则中得以体现，它可正式陈述如下：

从神经系统的输入层观测到的随机向量 $\mathbf{X}$ 到系统的输出层得到的随机向量 $\mathbf{Y}$ 之间的变换应该这样选择，这种变换使得输出层神经元的活动共同最大化关于输入层神经元的活动的信息。最大化的目标函数是向量 $\mathbf{X}$ 和 $\mathbf{Y}$ 之间的互信息 $I(\mathbf{Y};\mathbf{X})$。

最大互信息原则提供一个解决如图 10-2a 所描述的信息传输系统自组织的数学框架，它独立于实现它所使用的规则。同样，这个原则也可以看作信道容量这个概念在神经网络中的对应物，信道容量定义为通过一个通信信道的信息传输率的 Shannon 极限。

接下来，我们给出两个涉及有噪声的单神经元的例子说明最大互信息原则的应用。在一个例子中噪声出现在输出端，而在另一个例子中噪声出现在输入端。

例 10.4 被处理噪声破坏的单神经元 考虑线性神经元的简单情形，假设系统从 m 个源节点接受输入。令该神经元的输出中出现处理噪声，可表示为

499

$$Y = \left(\sum_{i=1}^{m} w_i X_i\right) + \mathrm{N} \tag{10.46}$$

其中 w_i 为第 i 个突触权值，N 为处理噪声，如图 10-3 所示的模型。假设：

- 输出 Y 是一个以方差为 σ_Y^2 的 Gauss 随机变量；

- 处理噪声 N 也是一个 Gauss 随机变量，均值为 0，方差为 σ_N^2。
- 处理噪声 N 与输入向量的任何一个分量都不相关，也即

$$E[\mathrm{N}X_i] = 0 \qquad 对所有的\ i$$

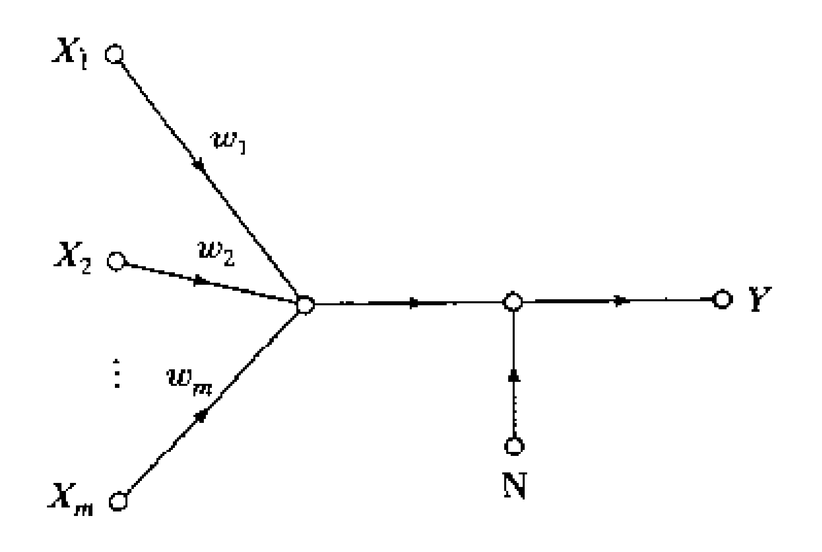


图 10-3 单个噪声神经元的信号流图

输出 Y 的高斯性可以用两种方法之一得到满足。输入 $X_1, X_2, \cdots, X_m$ 全部是 Gauss 分布的，再假设附加的噪声 N 也是高斯的，则 Y 的高斯性可以保证，这是由于一组 Gauss 分布的随机变量的加权和仍是高斯的。或者输入 $X_1, X_2, \cdots, X_m$ 是独立同分布的，在 m 很大的条件下利用中心极限定理它们的加权和趋于 Gauss 分布。

为了进行分析，我们首先注意在式(10.30)的第二行，输入向量 $\mathbf{X}$ 与输出变量 Y 之间的互信息 $I(Y;\mathbf{X})$ 是

$$I(Y;\mathbf{X}) = h(Y) - h(Y \mid \mathbf{X}) \tag{10.47}$$

根据式(10.46)，注意在已知输入向量 $\mathbf{X}$ 的情况下，输出 Y 的概率密度函数等于一个常数加上一个 Gauss 分布的随机变量的概率密度函数。因此，条件熵 $h(Y|\mathbf{X})$ 是由输出神经元传送的关于处理噪声 N 而不是向量 $\mathbf{X}$ 的"信息"。我们可以设置

$$h(Y \mid \mathbf{X}) = h(\mathrm{N})$$

因此式(10.47)可以重新简化为

$$I(Y;\mathbf{X}) = h(Y) - h(\mathrm{N}) \tag{10.48}$$

应用式(10.22)关于 Gauss 随机变量的微分熵到当前的问题，我们得到

$$h(Y) = \frac{1}{2}[1 + \log(2\pi\sigma_Y^2)] \tag{10.49}$$

和

$$h(\mathrm{N}) = \frac{1}{2}[1 + \log(2\pi\sigma_N^2)] \tag{10.50}$$

500

经过化简，将式(10.49)和式(10.50)代入式(10.48)得

$$I(Y;\mathbf{X}) = \frac{1}{2}\log\left(\frac{\sigma_Y^2}{\sigma_N^2}\right) \tag{10.51}$$

其中 σ_Y^2 依赖于 σ_N^2。

比值 σ_Y^2/σ_N^2 可看作信噪比。假设噪声方差 σ_N^2 为固定的约束条件，从(10.51)看出互信息 $I(Y;\mathbf{X})$ 通过神经元输出 Y 的方差 σ_Y^2 的最大化而成为最大化的。因此可以这样说，在一定的条件下，使神经元输出的方差最大化也就是使神经元的输出信号和它的输入之间的互信息最大化(Linsker, 1988a)。■

例 10.5 受附加输入噪声影响的单个神经元 假设噪声影响在每一个输入节点的突触末端的线性神经元的行为，如图 10-4 所示。根据这第二个噪声模型我们有

$$Y = \sum_{i=1}^{m} w_i(X_i + \mathrm{N}_i) \tag{10.52}$$

其中假设每个 N_i 是一个独立 Gauss 随机变量，其均值为 0，方差为 σ_N^2。我们可以将式(10.52)改写成类似式(10.46)的形式：

$$Y=\left(\sum_{i=1}^{m}w_iX_i\right)+\mathrm{N}'$$

其中 N′是噪声分量的组合，定义为

$$\mathrm{N}'=\sum_{i=1}^{m}w_i\mathrm{N}_i$$

噪声 N′是一个 Gauss 分布，其均值为 0，方差为所有独立噪声分量方差的加权和；即是

$$\sigma_{\mathrm{N}'}^2=\sum_{i=1}^{m}w_i^2\sigma_{\mathrm{N}}^2$$

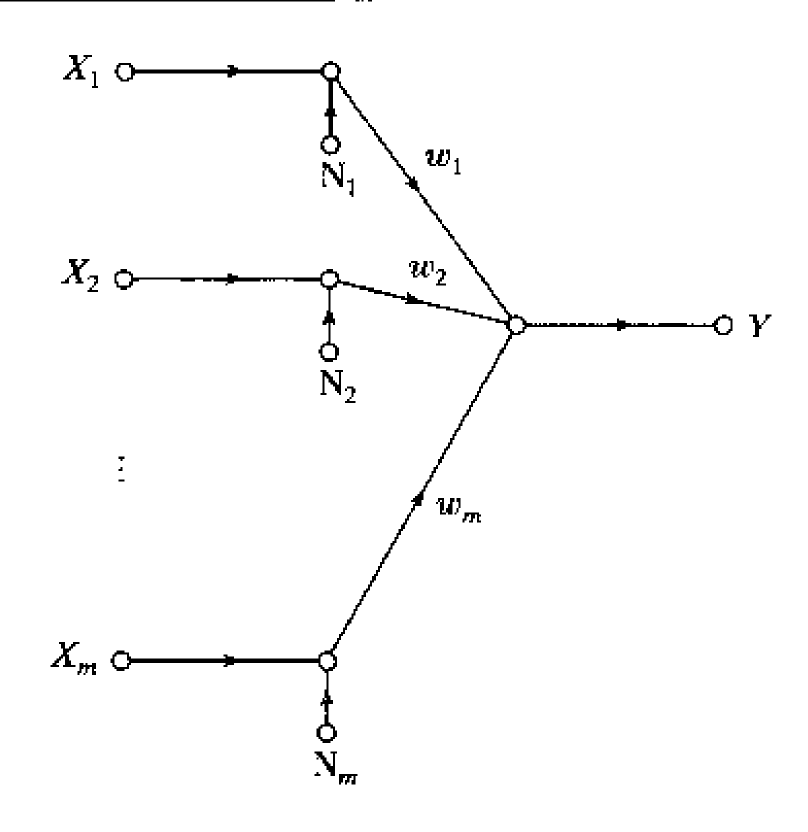


图 10-4 另一个噪声模型

501 与前类似，我们假设神经元的输出变量 Y 是方差为 σ_Y^2 的 Gauss 分布。Y 和 $\mathbf{X}$ 之间的互信息 $I(Y;\mathbf{X})$同样由式(10.47)给出。但是，这一次条件熵 $h(Y|\mathbf{X})$定义如下：

$$\begin{aligned}h(Y\mid\mathbf{X})&=h(\mathrm{N}')\\&=\frac{1}{2}(1+2\pi\sigma_{\mathrm{N}'}^2)\\&=\frac{1}{2}\left[1+2\pi\sigma_{\mathrm{N}}^2\sum_{i=1}^{m}w_i^2\right]\end{aligned}\tag{10.53}$$

这样，将式(10.49)和(10.53)代入式(10.47)并简化，可得(Linsker，1988a)

$$I(Y;\mathbf{X})=\frac{1}{2}\log\left(\frac{\sigma_Y^2}{\sigma_{\mathrm{N}}^2\sum_{i=1}^{m}w_i^2}\right)\tag{10.54}$$

在约束 σ_{N}^2 保持一个常量条件下，$I(Y;\mathbf{X})$的最大化就是比值 $\sigma_Y^2/\sum_{i=1}^{m}w_i^2$ 的最大化，其中 σ_Y^2 是 w_i 的函数。 ■

我们可从例 10.4 和例 10.5 推出什么结论 首先，从给出的两个例子可以看出，应用最大熵原则的结果依赖于问题。对于给定噪声方差 σ_{N}^2，最大化互信息 $I(Y;\mathbf{X})$和应用于图10-3的模型输出的方差之间的等价，并不能直接转到图 10-4 的模型。只有当对图 10-4 的模型加上$\sum_i w_i^2=1$ 的约束时，图 10-4 和图 10-3 所代表的模型才有相似的行为。

一般说来，确定输入向量 $\mathbf{X}$ 与输出向量 $\mathbf{Y}$ 的互信息 $I(\mathbf{Y};\mathbf{X})$是一件很困难的事。在例 10.4 和例 10.5 中，为了数学上分析的方便，我们假设系统噪声分布是一个或多个噪声源的多元 Gauss 分布。这个假设需要说明其合理性。

当采用 Gauss 噪声模型时，本质上是采用互信息的一个替代，其计算的前提是神经元的输出向量 $\mathbf{Y}$ 是一个均值向量和协方差矩阵都与实际情况相同的多维 Gauss 分布。在 Linsker (1993)中，利用 Kullback-Leibler 散度提供对于这种条件下的替代互信息的一个原则性理由，这些都假设网络已经存储关于输出向量 $\mathbf{Y}$ 的均值向量和协方差矩阵而不包含更高阶统计。

最后，在例 10.4 和例 10.5 给出的分析情况只是对于一个神经元进行的。有意这样做是意识到：为了最大互信息原则在数学上易于处理，最优化应该在局部神经元级进行。这种优化符合自组织的本质。

例 10.6 在例 10.4 和例 10.5 中，考虑了带有噪声的神经元。在本例中我们研究一个无

502 噪声的网络，它将任意分布的随机向量 $\mathbf{X}$ 变换为新的随机向量 $\mathbf{Y}$。注意 $I(\mathbf{X};\mathbf{Y})=I(\mathbf{Y};\mathbf{X})$，

并且在这里展开式(10.28)，可以将输入 **X** 和输出 **Y** 之间的互信息表达为

$$I(\mathbf{Y};\mathbf{X}) = H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X})$$

其中 $H(\mathbf{Y})$是 **Y** 的熵，$H(\mathbf{Y}|\mathbf{X})$是在给定 **X** 的条件下 **Y** 的条件熵。假设从 **X** 到 **Y** 的映射是无噪声的，条件熵取其最小的可能值：它发散到 $-\infty$。这是由于在 10.2 节讨论的连续随机变量熵的微分特性的必然结果。但是，当我们考虑互信息 $I(\mathbf{Y};\mathbf{X})$对参数化映射网络的权值矩阵 **W** 的梯度时，这个困难并不造成什么后果。特别是，我们可以写成

$$\frac{\partial I(\mathbf{Y};\mathbf{X})}{\partial \mathbf{W}} = \frac{\partial H(\mathbf{Y})}{\partial \mathbf{W}} \tag{10.55}$$

因为条件熵与 **W** 独立。式(10.55)表明，对于一个无噪声映射网络，最大化输出 **Y** 的熵就等于最大化 **Y** 和网络输入 **X** 之间的互信息 $H(\mathbf{X};\mathbf{Y})$，都是关于映射网络权矩阵 **W** 求最大化(Bell and Sejnowski, 1995)。 ■

10.8 最大互信息和冗余减少

在 Shannon 的信息论框架中，序和结构代表冗余，它减少接受方对信息分辨的不确定性。在固有过程中我们拥有的序和结构越多，则观察这个过程我们获得的信息量就越少。例如考虑高度结构化和冗余的序列 *aaaaaa*。一旦得到第一个样本 *a*，则我们就可以立即知道其余后面五个都是一样的 *a*。这样的一个序列所传递的信息的极限是单个符号传递的信息量。换句话说，样本序列的冗余越大，从环境中获取的信息内容也就越少。

从互信息 $I(\mathbf{Y};\mathbf{X})$的定义，我们知道这是对一个系统在已知输入为 **X** 时，对输出 **Y** 的不确定性的度量。最大互信息的方法是使互信息 $I(\mathbf{Y};\mathbf{X})$最大，其结果是我们在观测到输入为 **X** 时，对系统输出 **Y** 增加确定性。考虑到前面提到的信息与冗余之间的关系，因此我们可以说，最大互信息原则导致与在输入 **X** 中的冗余比较而言减少输出 **Y** 中的冗余。

噪声的出现是推动使用冗余以及相异性(diversity)相关方法的一个因素(Linsker, 1988a)。当输入信号的附加性噪声很高时，我们可以利用冗余来减少噪声的效果。在这种环境下，输入信号之间的更多(相关)分量都由处理器组合起来，以提供输入的精确表示。同样，当输出端的噪声(即处理器噪声)很高时，给出更多的输出分量以提供冗余信息。在处理器输出端观测到的相互独立的属性也相应地减少了，但各个属性表示的精确度反而提高了。因此我们可以这样说：*高水平的噪声有利于表示的冗余*。但是，*当噪声水平很低时，表示的相异性比冗余更有利*。我们用相异性表示处理器产生两个或多个具有不同性质的输出。习题 10.6 讨论的冗余/相异性的折中是由最大互信息观点得来的。值得一提的冗余/相异性折中与第 2 章提到的偏置/方差折中是类似的。 503

感知系统建模

自从信息论的早期，就提出了感觉消息(刺激)的冗余对感知理解非常有用(Attneave, 1954; Barlow, 1959)。感觉消息的冗余提供了人脑建立其周围环境的"认知映射"或"工作模型"(Barlow, 1989)。在感觉消息中规则必须以某种方式被人脑编码，使它知道什么经常发生。但是，*冗余减少*是 Barlow 假设的特定形式。这个假设说早期处理的目的是将高冗余的感觉输入转化成更有效的*析因码*(factorial code)。换句话说，在输入条件下使神经元输出统计独立。

受 Barlow 假设的启发，Atick and Redlich(1990)提出把*最小冗余原则*作为如图 10-5 所示的

感知系统的信息论模型的基础。系统的由三个部分组成：输入通道，重编码系统，输出通道。输入通道的输出可以表示为

$$\mathbf{X} = \mathbf{S} + \mathbf{N}_1$$

其中 $\mathbf{S}$ 是输入通道接收到的理想信号，$\mathbf{N}_1$ 假设为输入中所有噪声的源。随后信号 $\mathbf{X}$ 被线性矩阵算子 $\mathbf{A}$ 变换(重编码)，然后通过视觉神经或输出通道传输，产生输出 $\mathbf{Y}$，表示为

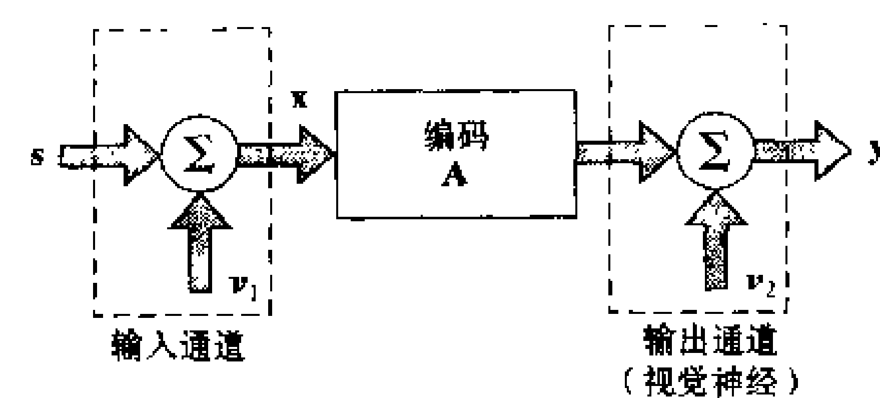


图 10-5　感知系统模型、信号向量 $\mathbf{s}$ 和噪声向量 $\mathbf{v}_1$ 和 $\mathbf{v}_2$ 分别是随机向量 $\mathbf{S}$，$\mathbf{N}_1$ 和 $\mathbf{N}_2$ 的值

$$\mathbf{Y} = \mathbf{AX} + \mathbf{N}_2$$

其中 $\mathbf{N}_2$ 表示后编码本身的噪声。在 Atick 和 Redlich 的方法中，观察到达视网膜的光信号包含一些非常有用的高冗余形式的感觉信息。进一步假设在信号沿视觉神经发送以前视网膜信号处理的目的就是减少或消除由于互相关性和噪声所带来的数据冗余。为了定量地描述这种观点，一个冗余度度量定义如下：

$$R = 1 - \frac{I(\mathbf{Y};\mathbf{S})}{C(\mathbf{Y})} \tag{10.56}$$

其中 $I(\mathbf{Y};\mathbf{S})$是 $\mathbf{Y}$ 和 $\mathbf{S}$ 之间的互信息，$C(\mathbf{Y})$是视觉神经(输出通道)的信道容量。式(10.56)的合理性基于人脑感兴趣的信息是理想的输入信号 $\mathbf{S}$，但是信息必须经过的物理信道实际上是视觉神经。假设在感知系统完成的输入与输出映射之间没有维数减少，这意味着 $C(\mathbf{Y}) > I(\mathbf{Y};\mathbf{S})$。要求找到一个输入－输出映射(即矩阵 $\mathbf{A}$)使冗余度度量 R 达到最小且满足不丢失信息的约束，可以表示为

$$I(\mathbf{Y};\mathbf{X}) = I(\mathbf{X};\mathbf{X}) - \varepsilon$$

其中 ε 是一个很小正参数。信道容量 $C(\mathbf{Y})$定义为保持平均输入能量固定的条件下和对所有应用于它的输入的概率分布，可能流过视觉神经的最大信息率。

当信号向量 $\mathbf{S}$ 和输出向量 $\mathbf{Y}$ 有相同的维数和系统存在噪声时，最小冗余度原则和最大互信息原则是数学上等价的，只要假设在两种情况下输出神经元计算能力的约束相同。具体地，假设根据图 10-5 的模型中信道容量的度量取决于每一个神经元输出的动态范围。那么，根据最小冗余度原则，对于一个给定的允许信息丢失，以及从而对于一个给定的 $I(\mathbf{Y};\mathbf{S})$，需要最小化的量定义为

$$1 - \frac{I(\mathbf{Y};\mathbf{S})}{I(\mathbf{S})}$$

因此，这样最小化的量本质上为

$$F_1(\mathbf{Y};\mathbf{S}) = C(\mathbf{Y}) - \lambda I(\mathbf{Y};\mathbf{S}) \tag{10.57}$$

另一方面，根据最大互信息原则，在图 10-5 的模型中需要最大化的量为

$$F_2(\mathbf{Y};\mathbf{S}) = I(\mathbf{Y};\mathbf{S}) + \lambda C(\mathbf{Y}) \tag{10.58}$$

虽然函数 $F_1(\mathbf{Y};\mathbf{S})$和 $F_2(\mathbf{Y};\mathbf{S})$并不相同，但是它们的最优化产生相同的结果：它们都是 Lagrange 乘子法的公式，仅仅是 $I(\mathbf{Y};\mathbf{S})$和 $C(\mathbf{Y})$简单地互换了角色。

从这些讨论中注意到这样一个重要的观点：虽然公式不同，但是这两个信息论的原则产生相似的结果。总的来说，一个神经网络输入和输出之间的互信息的最大化确实可以导出冗

余削减[9]。

10.9 空间相干特征

在 10.6 节中提出的最大互信息原则，主要应用于如图 10-2a 所示的情况下，神经系统的输出向量 **Y** 和输入向量 **X** 之间的互信息 $I(\mathbf{Y};\mathbf{X})$ 作为一个求最大值的目标函数。在术语上作适当改变，我们可以将其扩展到自然景物图像的无监督处理中(Becker and Hinton,1992)。一个未处理的图像的像素，虽然形式很复杂，但是包含我们感兴趣的景物的丰富信息。特别是，每个像素的密集度受内在参数的影响，例如深度、反射、表面方向和背景噪声以及照明度。目的就是设计一个自组织系统，能够学习将这种复杂的信息编码成一种简单的形式。更具体一点，目标就是从这个图像中提取能够展现该图像空间相干的高阶特征，使得在图像的空间局部区域的信息表示很容易产生邻近区域的信息表示；区域是指图像中的一组像素的集合。这种描述的情况属于图 10-2b 的场景。

因此我们可以将最大互信息原则的第一个变体[10]说明如下(Becker,1996; Becker and Hinton,1992)：

两个向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$(代表一个神经系统相邻的无重叠的图像区域)的变换应该如此选择，使得输入 $\mathbf{X}_a$ 对应的纯量输出 Y_a 最大化输入 $\mathbf{X}_b$ 对应的纯量输出 Y_b 的信息。最大化的目标函数就是输出 Y_a 和 Y_b 之间的互信息 $I(Y_a;Y_b)$。

我们称此为最大互信息原则的变体，意思是指它并不和最大互信息原则相等价或能够从其推导出来，但必定以相似的精神起作用。

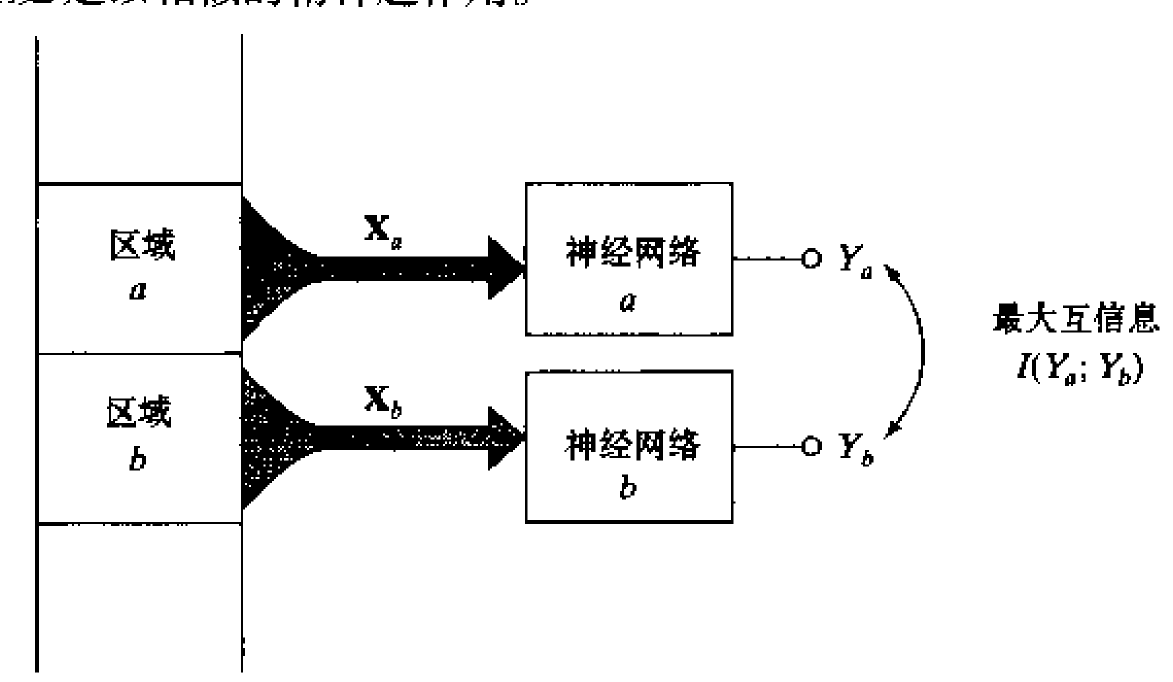


图 10-6 按照最大互信息的第一个变体处理图像的两个邻近区域

506

进一步我们考虑图 10-6 所示的情况，有两个神经网络(模型) a 和 b，分别接受输入为 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，来自同一图像中相邻的不重叠区域，各自的纯量输出分别是 Y_a 和 Y_b。令 S 表示 Y_a 和 Y_b 中共同信号分量，它是原始图像的两个相关区域的空间相干性的表示。我们可以将 Y_a 和 Y_b 看成共同信号 S 的带噪声形式，表示为

$$Y_a = S + N_a \tag{10.59}$$

和

$$Y_b = S + N_b \tag{10.60}$$

N_a 和 N_b 是加性噪声分量，假设为统计独立的零均值 Gauss 分布随机变量。信号分量 S 也假

设为Gauss分布的。根据式(10.59)和式(10.60)，在图10-6中假设模块a和b彼此相容。

利用式(10.30)的最后一行，Y_a和Y_b的互信息定义为

$$I(Y_a;Y_b) = h(Y_a) + h(Y_b) - h(Y_a,Y_b) \tag{10.61}$$

根据式(10.22)关于Gauss随机变量的微分熵，Y_a的微分熵$h(Y_a)$为

$$h(Y_a) = \frac{1}{2}[1 + \log(2\pi\sigma_a^2)] \tag{10.62}$$

其中σ_a^2是Y_a的方差。同理得Y_b的微分熵为

$$h(Y_b) = \frac{1}{2}[1 + \log(2\pi\sigma_b^2)] \tag{10.63}$$

其中σ_b^2是Y_b的方差。至于联合微分熵$h(Y_a, Y_b)$，我们利用式(10.24)得

$$h(Y_a,Y_b) = 1 + \log(2\pi) + \frac{1}{2}\log|\det(\mathbf{\Sigma})| \tag{10.64}$$

2×2的矩阵$\mathbf{\Sigma}$是Y_a和Y_b的协方差矩阵，定义为

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_a^2 & \rho_{ab}\sigma_a\sigma_b \\ \rho_{ab}\sigma_a\sigma_b & \sigma_b^2 \end{bmatrix} \tag{10.65}$$

其中ρ_{ab}是Y_a和Y_b的相关系数；也就是

$$\rho_{ab} = \frac{E[(Y_a - E[Y_a])(Y_b - E[Y_b])]}{\sigma_a\sigma_b} \tag{10.66}$$

所以矩阵$\mathbf{\Sigma}$的行列式为

$$\det(\mathbf{\Sigma}) = \sigma_a^2\sigma_b^2(1 - \rho_{ab}^2) \tag{10.67}$$

并且我们可以将式(10.64)重写为

507

$$h(Y_a,Y_b) = 1 + \log(2\pi) + \frac{1}{2}\log[\sigma_a^2\sigma_b^2(1 - \rho_{ab}^2)] \tag{10.68}$$

将式(10.62)，(10.63)和式(10.68)代入式(10.61)，并化简得

$$I(Y_a;Y_b) = -\frac{1}{2}\log(1 - \rho_{ab}^2) \tag{10.69}$$

从式(10.69)我们立即推出，最大化互信息$I(Y_a;Y_b)$等价于最大化相关系数ρ_{ab}。这从直观上看也是满足的。注意，由ρ_{ab}定义，$|\rho_{ab}|\leqslant 1$。

最大化$I(Y_a;Y_b)$可以看作统计学中求标准相关的非线性推广(Becker and Hinton, 1992)。给定两个输入向量(刺激)$\mathbf{X}_a$和$\mathbf{X}_b$(不必有相同的维数)，和相应的有两个权向量$\mathbf{w}_a$和$\mathbf{w}_b$，标准相关分析的目的就是指找到一个线性组合$Y_a = \mathbf{w}_a^T\mathbf{X}_a$和$Y_b = \mathbf{w}_b^T\mathbf{X}_b$，使它们之间的相关性最大(Anderson, 1984)。最大化$I(Y_a;Y_b)$为标准相关分析的非线性推广，是由于图10-6中神经网络内嵌模块设计的非线性。

在Becker and Hinton(1992)中，演示了通过最大互信息$I(Y_a;Y_b)$可以从一个随机体视点图中提取体视不均衡性(深度)。这是一个很困难的特征提取问题，不能由一个一层或线性神经元网络来解决。

10.10　空间非相干特征

在前面一节里我们讨论了一个无监督的图像处理过程，它从一个图像中提取空间相干特

征。现在我们将讨论与那里相反的问题。具体地说，考虑图 10-2c，其中目的是增强从两个不同图像中抽取相应区域的空间差异。在图 10-2b 中，我们是求模块输出间的互信息最大化，在图 10-2c 中我们做相反的工作。

因此我们可以将最大互信息原则的第二个变体，陈述如下(Ukrainec and Haykin, 1992, 1996)：

从两幅不同图像对应的区域得到的数据作为两个输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，神经系统对它们的变换的选择应该使得输入 $\mathbf{X}_a$ 对应的系统纯量输出 Y_a 关于输入 $\mathbf{X}_b$ 对应的系统纯量输出 Y_b 信息最小。最小化的目标函数是输出 Y_a 和 Y_b 之间的互信息 $I(Y_a;Y_b)$。

同样在这里我们称之为最大互信息原则的变体，意思是指它并不和最大互信息原则等价或能够从其推导出来，但必定以相似的精神起作用[11]。

最大互相信息原则的第二种变体在雷达偏振测定(radar polarimetry)方面有所应用。雷达 508 监视系统产生一对(或更多)我们感兴趣的环境的图像，利用在一个偏振方向上传送，在相同或不同偏振方向接收得到反向散射。偏振可以在垂直方向，也可以在水平方向上。例如，我们可能有两幅雷达图像，一幅图像代表相同方向(水平 - 水平)的偏振，而另一幅为交叉方向(水平 - 垂直)的偏振。这样的应用由 Ukrainec and Haykin(1992, 1996)提出，属于在一个双偏振雷达系统中的偏振目标增强。研究中雷达景物的采样描述如下。在一个非相干雷达以水平偏振方式传播，在垂直和水平偏振频道接收雷达返回。感兴趣的目标就是设计一个协件偏振扭曲反射器来将偶然偏振旋转 90 度。在普通的雷达系统操作中，这样一个目标的探测是非常困难的，既因为雷达系统的缺陷也因为地面目标会发生意想不到的偏振，并反射回来产生杂波(clutter)。我们发现需要用一个非线性映射来解释普通雷达返回结果的非 Gauss 分布。目标增强问题变为涉及约束二次函数最小化的求解问题。最终结果是一个处理后的交叉偏振图像，它在目标可见度方面表现出极大的提高，而且远比我们应用诸如主分量分析之类的线性技术得到的效果要好得多。因为模型无关的概率密度函数估计是一个计算量非常大的工作，所以 Ukraire 和 Haykin 提出的模型对变换后的数据假设是 Gauss 统计分布的。两个 Gauss 变量 Y_a 和 Y_b 的互信息由式(10.61)定义。为了学习两个模型的突触权值，采用了变通的方法。要求是抑制雷达杂波，对水平偏振和垂直偏振的雷达图像这是常见的。为了满足该要求，最小化互信息 $I(Y_a;Y_b)$，满足下面加在权值向量的约束条件：

$$P = (\mathrm{tr}[\mathbf{W}^T\mathbf{W}] - 1)^2 \tag{10.70}$$

其中 $\mathbf{W}$ 是网络总的权值矩阵，tr[.]是括号内矩阵的迹。如果

$$\nabla_{\mathbf{W}} I(Y_a;Y_b) + \lambda \nabla_{\mathbf{W}} P = 0 \tag{10.71}$$

成立，我们可以得到一个稳定点，其中 λ 是拉格朗日乘子。利用拟牛顿最优化程序寻找最小值。在第 4 章讨论拟牛顿方法。

图 10-7 显示 Ukrairec and Haykin(1992, 1996)所用的神经网络结构。对每个模型选择一个 Gauss 径向基函数网络(RBF)，这是因为它可以提供一系列的固定基函数的好处(即，有一个非自适应隐藏层)。输入数据在基函数上展开，然后通过线性权值层相结合；在图 10-7 中的虚线代表两个模块间的交叉耦合连接。Gauss 函数的中心在区间内均匀选择以便能完整覆盖全部输入区域，它们的宽度选择应用启发式规则。图 10-8a 显示一个在安大略湖岸边的一个公园的水平极化和垂直极化的雷达图像。每一幅图像的范围坐标是沿水平轴的，从左到右 509

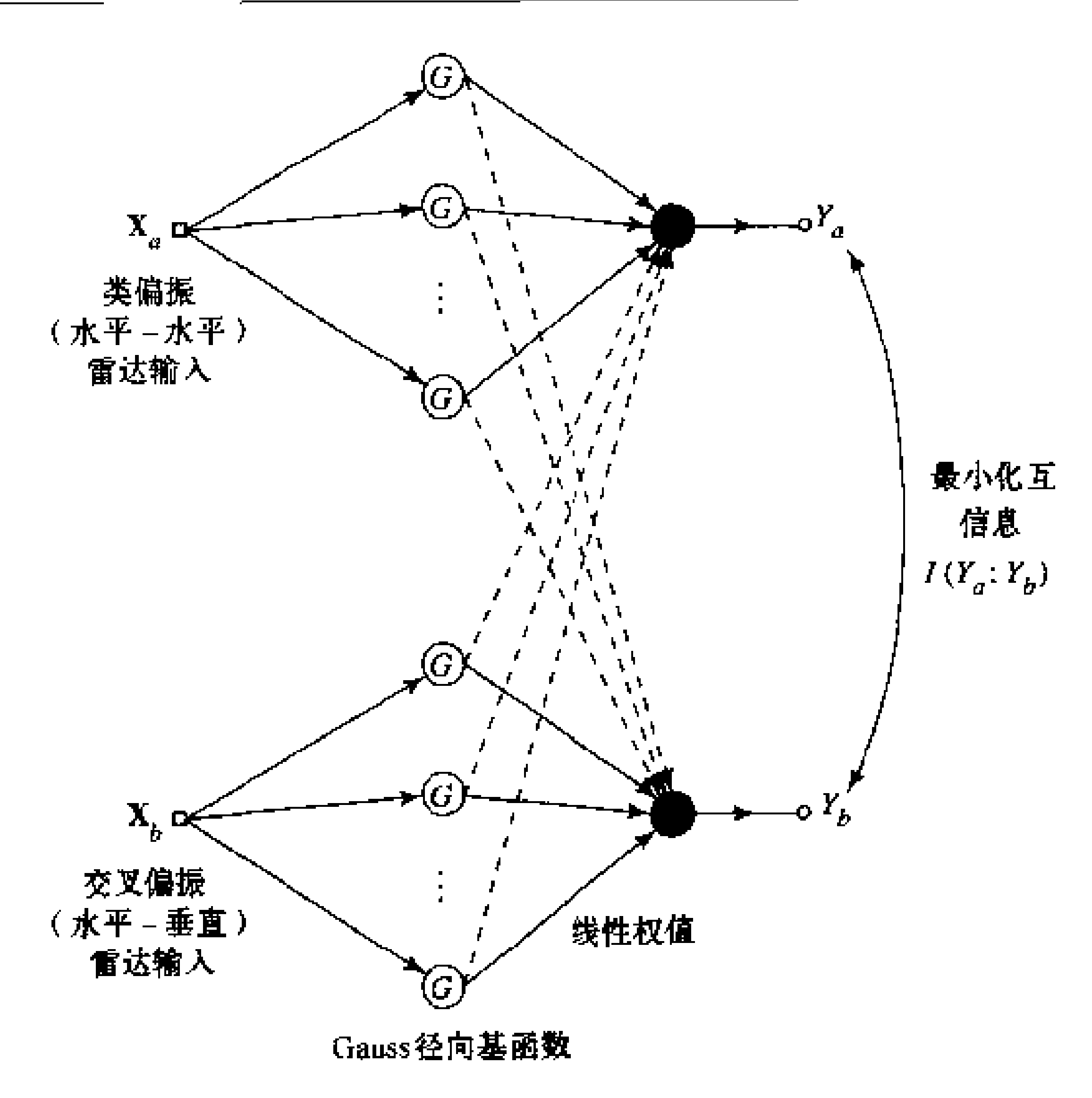


图 10-7 神经处理器框图，它的目标是利用一对偏振测定的非相干雷达输入抑制背景杂波；杂波抑制由最小化两个模型输出的互信息来达到

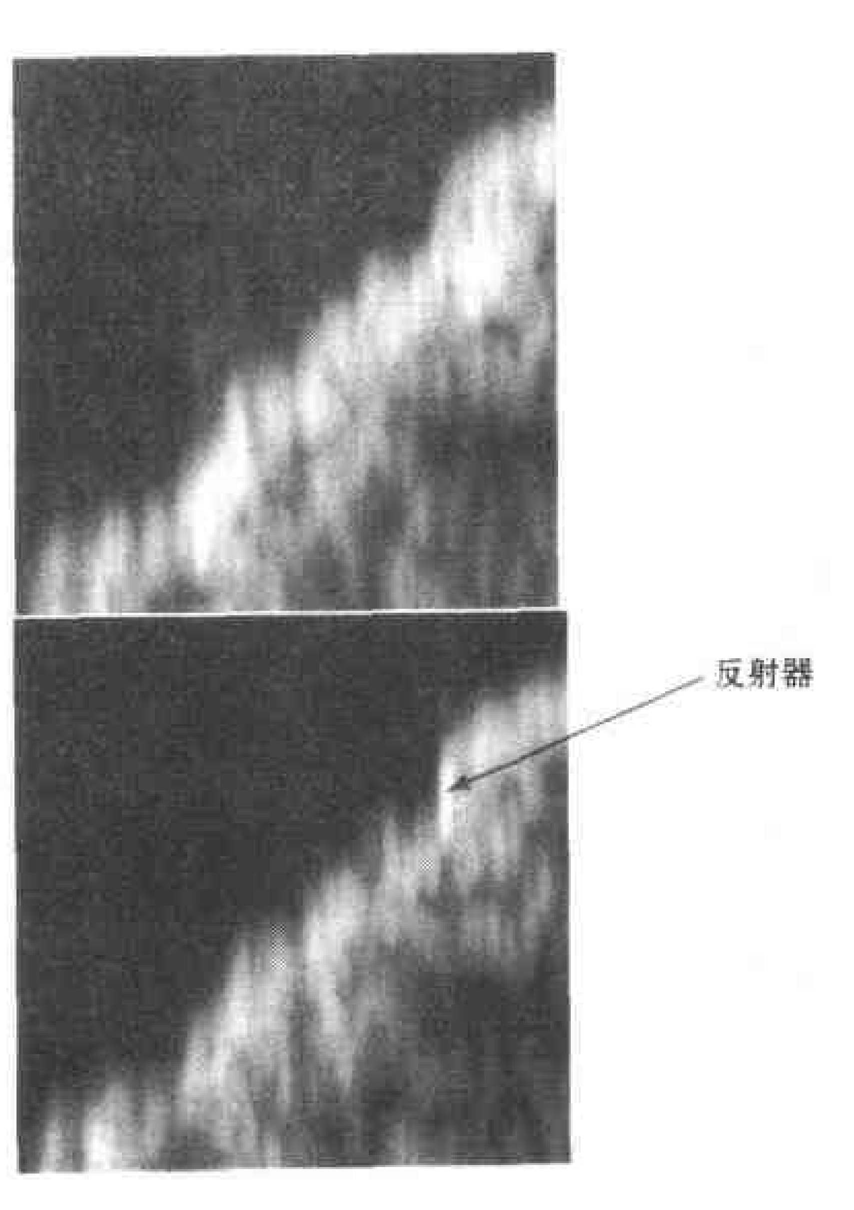


图 10-8 a)未处理的 B－扫描雷达图像(方位角和范围对比)，水平－水平偏振(上)和水平－垂直(下)偏振

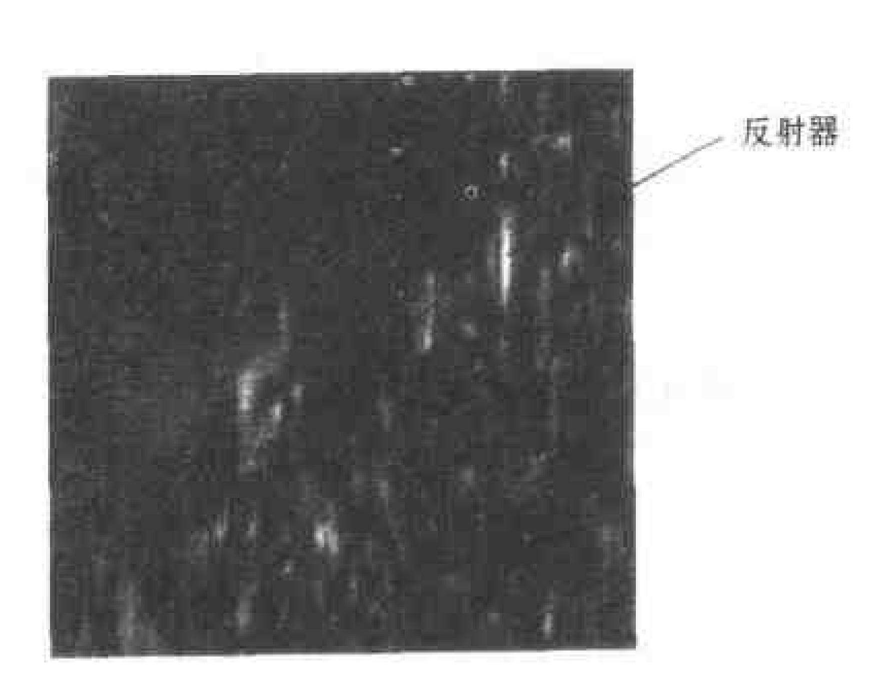


图 10-8 b)最小化图 10-8a)的两幅偏振雷达图像之间的互信息，计算得出的合成图像

递增；方位角坐标沿垂直轴。图 10-8b 显示采用最小化水平极化和垂直极化的雷达图像的互信息的组合图像。一个非常清晰的亮点在图像中可以看出来，它是根据雷达从放在湖边的一个协作偏振扭曲反射器返回的。这里描述的信息论模型的杂波压制的性能已超出了普通使用主分量分析方法利用投影的性能(Ukrainec and Haykin, 1992, 1996)[12]。

10.11 独立分量分析

现在我们将注意力集中在由图 10-2d 描述的最后场景。为了使那里陈述的信号处理问题更加具体化，考虑图 10-9 的方框图。操作从一个随机源向量 $\mathbf{U}(n)$开始，其定义为

$$\mathbf{U} = [U_1, U_2, \cdots, U_m]^T$$

其中 m 个分量是由一列独立源提供的。这里考虑时间序列；因而这里的 n 表示离散的时间。向量 $\mathbf{U}$ 应用到一个线性系统中，其输入输出之间的关系由一个非奇异的 $m \times m$ 的称为混合矩阵的 $\mathbf{A}$ 决定，结果是产生一个观察向量 $\mathbf{X}(n)$，它和 $\mathbf{U}(n)$关系如下(见图 10-10a)：

$$\mathbf{X} = \mathbf{AU} \tag{10.72}$$

510

其中 $\mathbf{X} = [X_1, X_2, \cdots, X_m]^T$。源向量 $\mathbf{U}$ 和混合矩阵 $\mathbf{A}$ 都是未知的，我们所知道的仅仅是观测向量 $\mathbf{X}$。给定 $\mathbf{X}$，问题是找到一个分离矩阵(demixing matrix)$\mathbf{W}$，使得可以从输出向量 $\mathbf{Y}$ 中恢复源向量 $\mathbf{U}$(见图 10-10b))，定义为

$$\mathbf{Y} = \mathbf{WX} \tag{10.73}$$

511

其中 $\mathbf{Y} = [Y_1, Y_2, \cdots, Y_m]^T$。通常假设源信号 $U_1, U_2, \cdots, U_m$ 是零均值的，这样可观测的 $X_1, X_2, \cdots, X_m$ 也是均值为零的信号。对分离器的输出 $Y_1, Y_2, \cdots, Y_m$ 也同样如此。

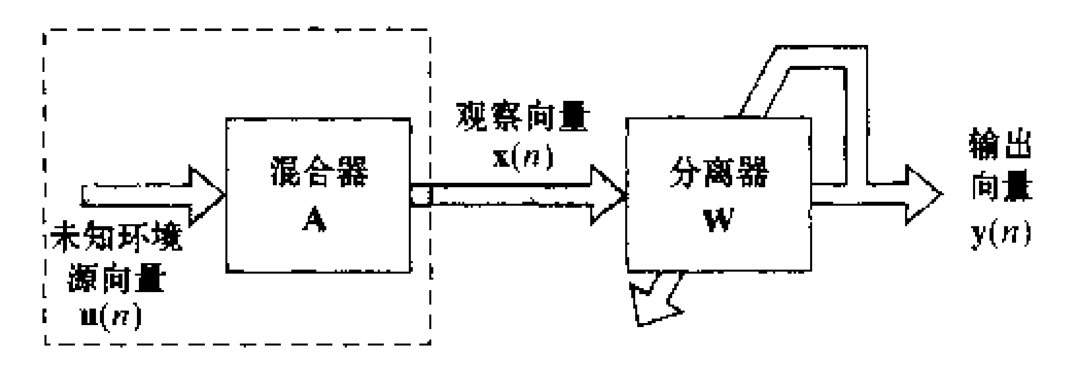


图 10-9 用于盲源分离问题的处理器方框图

向量 u，x 和 y 分别是随机向量 U，X 和 Y 的值

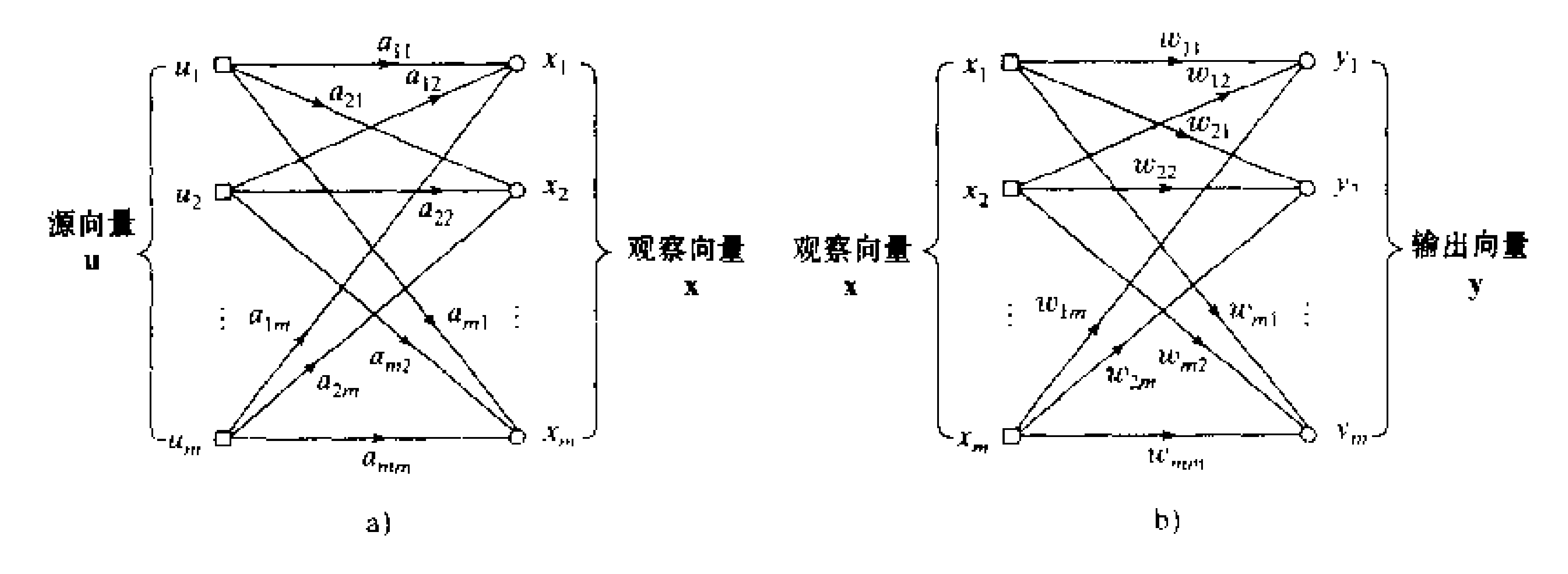


图 10-10 细节描述

a)混合矩阵 b)分离矩阵

[512] 我们可以定义盲源分离问题如下：

给定观测向量 $\mathbf{X}$ 的 N 个独立实现，找一个混合矩阵 $\mathbf{A}$ 的逆的估计。

源分离主要利用空间相异性，不同传感器提供的向量 $\mathbf{X}$ 的实现携带有源的不同混合。如果存在谱相异性，谱的相异性也可以被利用，但源分离问题的根本方法本质上是空间的：通过传感器而不是通过时间寻找结构(Cardoso,1998a)。

这种方法用于盲源分离问题是可行的，除了每个信号成分有一个任意尺度的变动，以及标号的置换。也就是说，可以找到一个分离矩阵 $\mathbf{W}$，它的每个列是混合矩阵 $\mathbf{A}$ 中的某列的置换和乘以一个比例系数。这种方法可以表达为

$$\mathbf{Y} = \mathbf{WX} = \mathbf{WAU} \to \mathbf{DPU}$$

的形式，其中 $\mathbf{D}$ 是一个非奇异对角矩阵，$\mathbf{P}$ 是一个置换矩阵。

在这里所描述的问题通常称为盲(信号)源分离问题[13]，其中使用"盲"这个术语是指用于恢复原始信号的仅有信息包含在观测向量 $\mathbf{X}$ 的实现中。在它的解答中内在的原则是*独立分量分析*(independent components analysis,ICA)(Comon,1994)，这可以看作是*主分量分析*(PCA)的一个拓展。而 PCA 强制到至多为二阶独立的，而且向量的方向限制为正交的，而 ICA 对于输出向量 $\mathbf{Y}$ 的单个分量限制为*统计独立*，并且没有正交性的限制。在实际中还应注意，独立分量分析的实现算法仅能够达到"尽可能统计独立"。

在多种应用中都出现盲源分离问题，包括以下几种情况：

- *语音分离*。这种应用中向量 $\mathbf{x}$ 由一些语音信号通过线性混合而成，要求就是将它们分离出来(Bell and Sejnowski,1995)。这种情况的困难形式，例如，出现在电视会议环境。
- *阵列天线处理*。在第二种应用中，向量 $\mathbf{x}$ 代表由一个雷达阵列天线产生的输出，它从未知方向的源发射一些偶然的窄带信号产生(Cardoso and Souloumia,1993; Swindlehurst et al.,1997)。这里的要求也是分离源信号。(对窄带信号我们是指一个带通信号，它的带宽比载波频率小。)
- *多传感器生物医学记录*。在这第三种应用中，向量 $\mathbf{x}$ 由用于监视生物信号的一些传感器产生的记录组成。例如，要求可能是从母亲的心跳中分离出胎儿的心跳(Cardoso,1998b)。
- [513] *金融市场数据分析*。在这种应用中，向量 $\mathbf{x}$ 由一系列不同的证券市场数据组成，要求抽取潜在的占优势的独立成分(Back and Weigend,1998)。

在这些应用中，盲源分离问题可能因为下列原因更复杂：可能存在未知传播延迟，它们的环境强加于源上的扩展滤波以及观测向量 $\mathbf{x}$ 难免混入的噪声。这些损害意味着(很不幸)在(10.72)所描述的瞬时混合的理想信号在现实世界上很少遇到。但在下面的讨论中，为了对盲源分离问题的基础理论有一个清楚的认识我们将忽略这些损害。

统计独立准则

由于对盲源分离输出向量 $\mathbf{Y}$ 的分量期望具有统计独立的性质，我们能用什么度量去测量独立性？一个明显的可能性是对组成输出向量 $\mathbf{Y}$ 的任意两个随机分量 Y_i 和 Y_j，利用它们的互信息 $I(Y_i;Y_j)$。在理想情况下，当 $I(Y_i;Y_j)$ 为零时，Y_i 和 Y_j 统计独立。因此这将意味

者对组成输出向量 **Y** 的任意两个随机变量 Y_i 和 Y_j，最小化它们的互信息 $I(Y_i;Y_j)$。这个目标等价于最小化下列两个分布的 Kullback-Leibler 散度：(1)概率密度函数 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$被 **W** 参数化；(2)相应的析因分布定义为

$$\tilde{f}_{\mathbf{Y}}(\mathbf{y},\mathbf{W}) = \prod_{i=1}^{m}\tilde{f}_{Y_i}(y_i,\mathbf{W}) \tag{10.74}$$

其中 $\tilde{f}_{Y_i}(y_i,\mathbf{W})$是 Y_i 的边缘概率密度函数。实际上(10.74)可以看作是加在学习算法上的约束，使算法对 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$与 $\tilde{f}_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$分开。我们可以将最大互信息原则的第三种变体陈述如下(Comon,1994)：

> 给定一个 $m\times 1$ 的向量 **X**，它表示 m 个独立源信号的一个线性组合。由神经系统将输入向量 **X** 变换为输出向量 **Y**，该变换应这样进行，使得参数化概率(记为 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$)与相应的析因分布 $\tilde{f}_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$之间的 Kullback-Leibler 散度关于未知参数矩阵 **W** 最小化。

这里所描述的用于问题的 Kullback-Leibler 散度在 10.5 节已经考虑。我们要找的公式由式(10.44)给出。应用该公式到目前这种情况，可以将 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$与 $\tilde{f}_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$的 Kullback-Leibler 散度表示为

$$D_{f\|\tilde{f}}(\mathbf{W}) = -h(\mathbf{Y}) + \sum_{i=1}^{m}\tilde{h}(Y_i) \tag{10.75}$$

514

其中 $h(\mathbf{Y})$是分离器输出的随机向量 **Y** 的熵，$\tilde{h}(Y_i)$是 **Y** 的第 i 个元素的边缘熵。Kullback-Leibler 散度 $D_{f\|\tilde{f}}$就是以后我们解决盲源分离问题的目标函数。

微分熵 $h(\mathbf{Y})$的确定

由式(10.73)给出输出向量 **Y** 与输入向量 **X** 有关，其中 **W** 是分离矩阵。根据式(10.18)，我们可以把 **Y** 的微分熵表示为

$$h(\mathbf{Y}) = h(\mathbf{WX}) = h(\mathbf{X}) + \log|\det(\mathbf{W})| \tag{10.76}$$

其中 $\det(\mathbf{W})$是 **W** 的行列式。

边缘熵 $\tilde{h}(Y_i)$的确定

为了求 Kullback-Leibler 散度 $D_{f\|\tilde{f}}$，我们也需要知道边缘熵 $\tilde{h}(Y_i)$。为了确定 $\tilde{h}(Y_i)$需要知道 Y_i 的边缘分布，这就要求累计随机向量 **Y** 除了 i 外的所有分量的作用。对于一个高维的向量 **Y** 来说，求 $\tilde{h}(Y_i)$要比求 $h(\mathbf{Y})$困难得多。根据随机变量 Y_i 的高阶矩我们推导出 $\tilde{h}(Y_i)$的一个近似表达式来克服这个困难。适当截断下面两个展开式中的一个可以完成这个任务：

- Edgeworth 级数(Comon,1991)
- Gram-Charlier 级数(Amari et al.,1996)

在本章中，我们将运用第二种方法。在注释[4]中给出 Gram-Charlier 级数的说明。在该注释中对 Edgeworth 级数也做了扼要描述。

具体地说，参数化的边缘概率密度函数 $\tilde{f}_{Y_i}(y_i,\mathbf{W})$的 Gram-Charlier 展开式表示为

$$\tilde{f}_{Y_i}(y_i,\mathbf{W}) = \alpha(y_i)\left[1+\sum_{k=3}^{\infty} c_k H_k(y_i)\right] \tag{10.77}$$

其中各项的定义如下：

1. 乘数因子 $\alpha(y_i)$是一个具有零均值和方差为 1 的归一化的 Gauss 随机变量的概率密度函数；即

$$\alpha(y_i) = \frac{1}{\sqrt{2\pi}} e^{-y_i^2/2}$$

2. $H_k(y_i)$是 Hermite 多项式。

3. 展开系数$\{c_k: k=3,4,\cdots\}$由随机变量 Y_i 的累计量定义。

在(10.77)中各项的自然顺序并不是 Gram-Charlier 级数中最好的。相反，下面括号中列出的项应组合在一起(Helstrom, 1968)：

$$k = (0),(3),(4,6),(5,7,9),\cdots$$

对于盲源分离问题，Gram-Charlier 级数中在 $k=(4,6)$截断时，对边缘概率函数 $\tilde{f}_{Y_i}(y_i)$的逼近就足够了。于是我们可以写成

515

$$\tilde{f}_{Y_i}(y_i) \approx \alpha(y_i)\left(1+\frac{\kappa_{i,3}}{3!}H_3(y_i)+\frac{\kappa_{i,2}^2}{4!}H_4(y_i)+\frac{(\kappa_{i,6}+10\kappa_{i,3}^2)}{6!}H_6(y_i)\right) \tag{10.78}$$

其中 $\kappa_{i,k}$是 Y_i 的第 k 阶累积量。令 $m_{i,k}$表示 Y_i 的第 k 阶矩，定义为

$$\begin{aligned} m_{i,k} &= E[Y_i^k] \\ &= E\left[\left(\sum_{k=1}^{m} w_{ik}X_i\right)^k\right] \end{aligned} \tag{10.79}$$

其中 X_i 是向量 $\mathbf{X}$ 的第 i 个元素，w_{ik}是权值矩阵 $\mathbf{W}$ 中的(i,k)元素。在此之前我们已经假设所有的 Y_i 的均值为零。相应地，我们有方差 $\sigma_i^2 = m_{i,2}$(即方差和均方值相等)，而且 Y_i 的 k 阶累积量同样如此：

$$\kappa_{i,3} = m_{i,3} \tag{10.80}$$

$$\kappa_{i,4} = m_{i,4} - 3m_{i,2}^2 \tag{10.81}$$

$$\kappa_{i,6} = m_{i,6} - 10m_{i,3}^2 - 15m_{i,2}m_{i,4} + 30m_{i,2}^3 \tag{10.82}$$

利用式(10.78)的逼近，$\tilde{f}_{Y_i}(y_i)$的算法给出如下：

$$\log\tilde{f}_{Y_i}(y_i) \approx \log\alpha(y_i) + \log\left(1+\frac{\kappa_{i,3}}{3!}H_3(y_i)+\frac{\kappa_{i,2}^2}{4!}H_4(y_i)+\frac{(\kappa_{i,6}+10\kappa_{i,3}^2)}{6!}H_6(y_i)\right) \tag{10.83}$$

为了继续进行，我们利用对数展开式

$$\log(1+y) \approx y - \frac{y^2}{2} \tag{10.84}$$

其中三阶和三阶以上的项都被省略了。

从前面的讨论，我们回忆计算 Y_i 的边缘熵的公式为(参看(10.43))

$$\tilde{h}(Y_i) = -\int_{-\infty}^{\infty} f_{Y_i}(y_i)\log f_{Y_i}(y_i)\,dy_i, \qquad i=1,2,\cdots,m$$

其中 m 是源的数目。利用式(10.78)，(10.83)和式(10.84)中的近似值，进行涉及 $\alpha(y_i)$和各种 Hermite 多项式 $H_k(y_i)$的积分，我们得到边缘熵的近似公式(Madhuaranth and Haykin, 1998)：

516

$$
\begin{aligned}
h(Y_i) \approx & \frac{1}{2}\log(2\pi e) - \frac{\kappa_{i,3}^2}{12} - \frac{\kappa_{i,4}^2}{48} - \frac{(\kappa_{i,6} + 10\kappa_{i,3}^2)^2}{1440} \\
& + \frac{3}{8}\kappa_{i,3}^2\kappa_{i,4} + \frac{\kappa_{i,3}^2(\kappa_{i,6} + 10\kappa_{i,3}^2)}{24} + \frac{\kappa_{i,4}^2(\kappa_{i,6} + 10\kappa_{i,3}^2)}{24} \\
& + \frac{\kappa_{i,4}(\kappa_{i,6} + 10\kappa_{i,3}^2)^2}{64} + \frac{\kappa_{i,4}^3}{16} + \frac{(\kappa_{i,6} + 10\kappa_{i,3}^2)^3}{432}
\end{aligned}
\tag{10.85}
$$

用式(10.76)和式(10.85)代入式(10.75)，我们得到目前问题的 Kullback-Leibler 散度：

$$
\begin{aligned}
D_{f\|\tilde{f}}(\mathbf{W}) \approx & - h(\mathbf{X}) - \log|\det(\mathbf{W})| + \frac{m}{2}\log(2\pi e) \\
& - \sum_{i=1}^{m}\Bigg(\frac{\kappa_{i,3}^2}{12} + \frac{\kappa_{i,4}^2}{48} + \frac{(\kappa_{i,6} + 10\kappa_{i,3}^2)^2}{1440} - \frac{3}{8}\kappa_{i,3}^2\kappa_{i,4} \\
& - \frac{\kappa_{i,3}^2(\kappa_{i,6} + 10\kappa_{i,3}^2)}{24} - \frac{\kappa_{i,4}^2(\kappa_{i,6} + 10\kappa_{i,3}^2)}{24} \\
& - \frac{\kappa_{i,4}(\kappa_{i,6} + 10\kappa_{i,3}^2)^2}{64} - \frac{\kappa_{i,4}^3}{16} - \frac{(\kappa_{i,6} + 10\kappa_{i,3}^2)^3}{432}\Bigg)
\end{aligned}
\tag{10.86}
$$

其中累积量都是权值矩阵 **W** 的函数。

激活函数

为了计算(10.86)中 Kullback-Leibler 散度，我们需要一个计算观测向量 **X** 的高阶累计量的自适应过程。问题是我们如何进行这些计算？记住导出式(10.86)近似公式的方法。它的导出是通过 Gram-Charlier 级数展开得到的，而且假设 Y_i 是零均值和方差为 1 的随机变量。零均值的假设是因为以前我们假定源信号为零均值的。至于方差为 1 的假设，要用到以下两种方法中的一种进行处理：

1. *约束方法*。在这种方法中，单位方差的假设用于计算对所有 i 的高阶累积量 $\kappa_{i,3}$，$\kappa_{i,4}$ 和 $\kappa_{i,6}$(Amari,1996)。不幸的是我们不能保证在计算过程中 Y_i 的方差(即 σ_i^2)是常数，不要说是 1 了。从式(10.81)和(10.82)的定义中注意 $\kappa_{i,4}$ 和 $\kappa_{i,6}$ 的估计依赖于 $\sigma_i^2 = m_{i,2}$。假设 $\sigma_i^2 = 1$，则导出 $\kappa_{i,4}$ 和 $\kappa_{i,6}$ 的估计有极大偏差，这将引起它们和 $\kappa_{i,3}$ 估计之间的错误关系。

2. *无约束方法*。在这种代替方法中，方差 σ_i^2 被看作是一个未知的时变参数，这也是与实际情况相符的(Madhuranath and Haykin,1998)。方差 σ_i^2 与 1 的偏离可以看作随机变量 Y_i 的一个比例变化。重要的是，导出的 $\kappa_{i,4}$ 和 $\kappa_{i,6}$ 的估计考虑到了 σ_i^2 是随时间变换的。在式(10.86)中的所有 3 个高阶累积量的估计还维持正确的关系。

517

在 Madhuranath and Haykin(1998)所作的盲源分离实验的研究报告表明，无约束方法产生的结果比约束方法的要好。在后面的讨论中我们使用无约束方法。

为了找到计算 **W** 的一个学习算法，我们要求式(10.86)对 **W** 的微分，从而对算法形成一个合适的激活函数。

令 A_{ik} 表示矩阵 **W** 的 ik 余子式。对 det(**W**)按 i 行进行拉普拉斯展开，可以写成(Wylie and Barrett,1982)

$$
\det(\mathbf{W}) = \sum_{k=1}^{m} w_{ik}A_{ik}, \qquad i = 1,2,\cdots,m \tag{10.87}
$$

其中 w_{ik} 是矩阵 $\mathbf{W}$ 的(i,k)元素。因此 $\det(\mathbf{W})$对 w_{ik} 求微分，得到

$$\frac{\partial}{\partial w_{ik}}\log(\det(\mathbf{W})) = \frac{1}{\det(\mathbf{W})}\frac{\partial}{\partial w_{ik}}\det(\mathbf{W}) = \frac{A_{ik}}{\det(\mathbf{W})} = (\mathbf{W}^{-T})_{ik} \tag{10.88}$$

其中 $\mathbf{W}^{-T}$ 是转置矩阵 $\mathbf{W}^{T}$ 的逆。在式(10.86)中其他项(依赖于 W)对 w_{ik} 求偏微分得到(参见式(10.80)至式(10.82))

$$\frac{\partial \kappa_{i,3}}{\partial w_{ik}} = 3E[Y_i^2 X_k]$$

$$\frac{\partial \kappa_{i,4}}{\partial w_{ik}} = 4E[Y_i^3 X_k] - 12m_{i,2}E[Y_i X_k]$$

$$\frac{\partial}{\partial w_{ik}}(\kappa_{i,6} + 10\kappa_{i,3}^2) = 6E[Y_i^5 X_k] - 30m_{i,4}E[Y_i X_k] - 60m_{i,2}E[Y_i^3 X_k] + 180m_{i,2}^2 E[Y_i X_k]$$

为了推导一个自适应算法，常用的方法是将期望用它们的瞬时值代替。因此在这三个等式中做如上的替换，我们得到下面的近似结果：

$$\frac{\partial \kappa_{i,3}}{\partial w_{ik}} \approx 3y_i^2 x_k \tag{10.89}$$

$$\frac{\partial \kappa_{i,4}}{\partial w_{ik}} \approx -8y_i^3 x_k \tag{10.90}$$

$$\frac{\partial}{\partial w_{ik}}(\kappa_{i,6} + 10\kappa_{i,3}^2) \approx 96y_i^5 x_k \tag{10.91}$$

在式(10.86)的表达式中对 w_{ik} 的导数用式(10.88)至式(10.91)替代，得到

$$\frac{\partial}{\partial w_{ik}}D_{f\|f}(\mathbf{W}) \approx -(\mathbf{W}^{-T})_{ik} + \varphi(y_i)x_k \tag{10.92}$$

其中的 $\varphi(y_i)$ 是学习算法的一个非单调激活函数，定义为(Madhuranath and Haykin, 1998)

518

$$\varphi(y_i) = \frac{1}{2}y_i^5 + \frac{2}{3}y_i^7 + \frac{15}{2}y_i^9 + \frac{2}{15}y_i^{11} - \frac{112}{3}y_i^{13} + 128y_i^{15} - \frac{512}{3}y_i^{17} \tag{10.93}$$

图 10.11 画出 $\varphi(y_i)$对 $y_i(-1<y_i<1)$的图形。图中包括分离器输出 y_i 的取值范围，这也是学习算法通常限制的范围。值得注意的是激活函数的斜率在$(-0.734, 0.734)$之间为正的；如同本节后面讨论的那样这是使算法稳定所必需的。

ICA 学习算法

学习算法的目的就是最小化概率密度函数 $\mathbf{Y}$ 和析因分布 Y_i，$i=1, 2, \cdots, m$ 之间的 Kullback-Leibler 散度。这个最小化可以运用梯度下降法实现，此时权值 w_{ik} 的调整定义为

$$\begin{aligned}\Delta w_{ik} &= -\eta\frac{\partial}{\partial w_{ik}}D_{f\|\tilde{f}} \\ &= \eta((\mathbf{W}^{-T})_{ik} - \varphi(y_i)x_k)\end{aligned} \tag{10.94}$$

其中 η 是学习率参数。

将(10.94)扩展到分离器的整个权值矩阵 $\mathbf{W}$，我们可以把适用于 $\mathbf{W}$ 调整的 $\Delta\mathbf{W}$ 表示为

519

$$\Delta\mathbf{W} = \eta(\mathbf{W}^{-T} - \boldsymbol{\varphi}(\mathbf{y})\mathbf{x}^T) \tag{10.95}$$

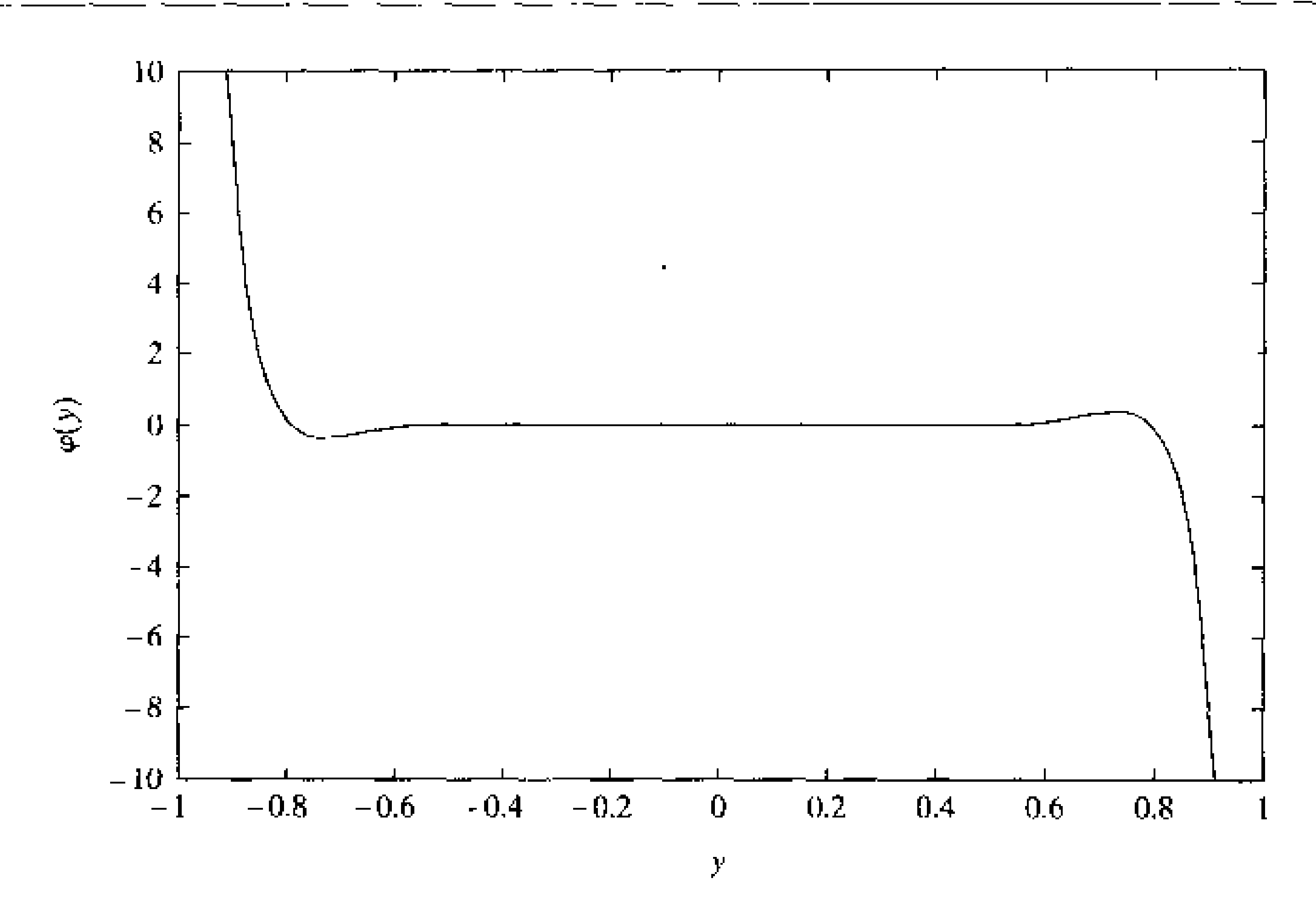


图 10-11 式(10.93)的激活函数 $\varphi(y)$

其中 $\mathbf{x}^T$ 是 $m\times 1$ 观测向量 $\mathbf{x}$ 的转置，并且

$$\boldsymbol{\varphi}(\mathbf{y}) = [\varphi(y_1),\varphi(y_2),\cdots,\varphi(y_m)]^T \tag{10.96}$$

在式(10.95)给出的 $\Delta\mathbf{W}$ 的公式中，注意到

$$\mathbf{y}^T = \mathbf{x}^T\mathbf{W}^T$$

我们可以将式(10.95)改写成等价形式

$$\Delta\mathbf{W} = \eta[\mathbf{I} - \boldsymbol{\varphi}(y)\mathbf{x}^T\mathbf{W}^T]\mathbf{W}^{-T} = \eta[\mathbf{I} - \boldsymbol{\varphi}(\mathbf{y})\mathbf{y}^T]\mathbf{W}^{-T} \tag{10.97}$$

其中 $\mathbf{I}$ 是单位矩阵。改变分离矩阵的更新公式表达为

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(n)[\mathbf{I} - \boldsymbol{\varphi}(\mathbf{y}(n))\mathbf{y}^T(n)]\mathbf{W}^{-T}(n) \tag{10.98}$$

其中参数都是以它们的时变形式给出。

等变化性质

盲源分离算法的目的是更新分离矩阵 $\mathbf{W}(n)$，以使输出向量

$$\mathbf{y}(n) = \mathbf{W}(n)\mathbf{x}(n) = \mathbf{W}(n)\mathbf{A}\mathbf{u}(n)$$

在统计意义下能够尽可能与初始源信号接近。具体地，假设一个全局系统表征矩阵 $\mathbf{C}(n)$是混合矩阵 $\mathbf{A}$ 和分离矩阵 $\mathbf{W}(n)$的乘积：

$$\mathbf{C}(n) = \mathbf{W}(n)\mathbf{A} \tag{10.99}$$

理想情况下，这个全局系统应该满足两个条件：

1. 负责调整 $\mathbf{C}(n)$的算法收敛到一个等于置换矩阵的最优值。
2. 算法本身可以写成

$$\mathbf{C}(n+1) = \mathbf{C}(n) + \eta(n)\mathbf{G}(\mathbf{C}(n)\mathbf{u}(n))\mathbf{C}(n) \tag{10.100}$$

其中 $\mathbf{G}(\mathbf{C}(n)\mathbf{u}(n))$是 $\mathbf{C}(n)\mathbf{u}(n)$的向量值函数。算法的性能完全由系统矩阵 $\mathbf{C}(n)$决定，而不是由混合矩阵 $\mathbf{A}$ 和分离矩阵 $\mathbf{W}(n)$单独决定。这样的自适应性系统就称为等变化的(Cardoso and Laheld, 1996)。

式(10.98)的自适应算法当然能够近似满足第一个条件。但是，正如它所表示的，第二个条件不能满足。为了说明这个问题，我们可以将式(10.98)重写成等价形式

$$\mathbf{C}(n+1)=\mathbf{C}(n)+\eta(n)\mathbf{G}(\mathbf{C}(n)\mathbf{u}(n))\mathbf{W}^{-T}(n)\mathbf{A} \tag{10.101}$$

其中

520

$$\mathbf{G}(\mathbf{C}(n)\mathbf{u}(n))=\mathbf{I}-\boldsymbol{\varphi}(\mathbf{C}(n)\mathbf{u}(n))(\mathbf{C}(n)\mathbf{u}(n))^T \tag{10.102}$$

式(10.98)的算法不能满足式(10.100)所描述的等变化条件，因为向量值函数 $\mathbf{G}(\mathbf{C}(n)\mathbf{u}(n))$ 后乘以 $\mathbf{W}^{-T}(n)\mathbf{A}$，在一般的条件下其值不等于 $\mathbf{C}(n)$。我们可以在他们之间插入一个矩阵 $\mathbf{W}^T(n)\mathbf{W}(n)$ 来矫正。项 $\mathbf{W}^T\mathbf{W}$ 由 $\mathbf{W}$ 和其转置组成，总是正定的。这也是为什么乘以 $\mathbf{W}^T\mathbf{W}$ 后不改变学习算法的最小值符号的原因。

重要的问题是：为了达到等变化条件所做的调整含义是什么？问题的答案就在于在参数空间中梯度下降是如何形成的。理想情况下，我们应该用目标函数 $D_{f\|g}(\mathbf{W})$ 的自然梯度[15]，利用通常梯度 $\nabla D_{f\|\tilde{f}}$ 定义为

$$\nabla^* D_{f\|\tilde{f}}(\mathbf{W})=(\nabla D_{f\|\tilde{f}}(\mathbf{W}))\mathbf{W}^T\mathbf{W} \tag{10.103}$$

通常梯度 $\nabla D_{f\|\tilde{f}}$ 由(10.92)定义。在隐含的意义下，梯度 $\nabla D_{f\|\tilde{f}}(\mathbf{W})$ 仅当参数空间 $\mathscr{W}=\{\mathbf{W}\}$ 是采用正交坐标系的欧几里德空间时为最优下降方向。在神经网络的典型情况中，参数空间 $\mathscr{W}$ 是在非正交坐标系中的。自然梯度 $\nabla^* D_{f\|\tilde{f}}(\mathbf{W})$ 在后一种情况下会产生最速下降，所以在构成盲源分离问题的随机算法时采用它替代通常梯度。为了使自然梯度空间可定义，必须满足两个条件：

1. 参数空间 $\mathscr{W}$ 是黎曼空间[16]。黎曼结构是一个具有正定度量 $\mathbf{W}$ 的微分流形。
2. 矩阵 $\mathbf{W}$ 是非奇异的(即可逆的)。

当前的问题对两个条件都满足。

以这种方式改写式(10.98)的算法，我们可以写为

$$\begin{aligned}\mathbf{W}(n+1)&=\mathbf{W}(n)+\eta(n)[\mathbf{I}-\boldsymbol{\varphi}(\mathbf{y}(n))\mathbf{y}^T](\mathbf{W}(n)\mathbf{W}^T(n))\mathbf{W}^{-T}(n)\\&=\mathbf{W}(n)+\eta(n)[\mathbf{I}-\boldsymbol{\varphi}(\mathbf{y}(n))\mathbf{y}^T(n)]\mathbf{W}(n)\end{aligned} \tag{10.104}$$

这导致盲源分离具有等方差(equivariance)性质。图10-12画出式(10.104)的信号流图。

为了使式(10.104)所描述的自适应算法对图10-9所描述的盲源分离问题得到正确结果，输出向量 $\mathbf{Y}$ 的所有分量必须满足下列两个要求：

- 用于计算非线性 $\varphi(\cdot)$ 的 Gram-Charlier 展开要包括足够多的项以便对边缘熵 $h(Y_i)$ 产生尽可能好的逼近；例如，式(10.93)的激活函数可以满足这个要求。
- 学习率 η 应足够的小，使得 Y_i 的累积量估计可靠。

稳定性分析

不对式(10.104)所描述的自适应算法进行稳定性分析，盲源分离问题的讨论是不完全的。在 Amari et al.(1997)，对任何激活函数 $\varphi(\cdot)$ 给出这个算法的一般性的稳定性分析。在算法渐进收敛于一个希望的平衡点的意义下进行分析，在希望的平衡点盲源的成功分离是有保证的。

521

式(10.104)是基于自然梯度的盲源分离算法的离散时间描述。为了稳定性分析，算法改为连续时间的形式

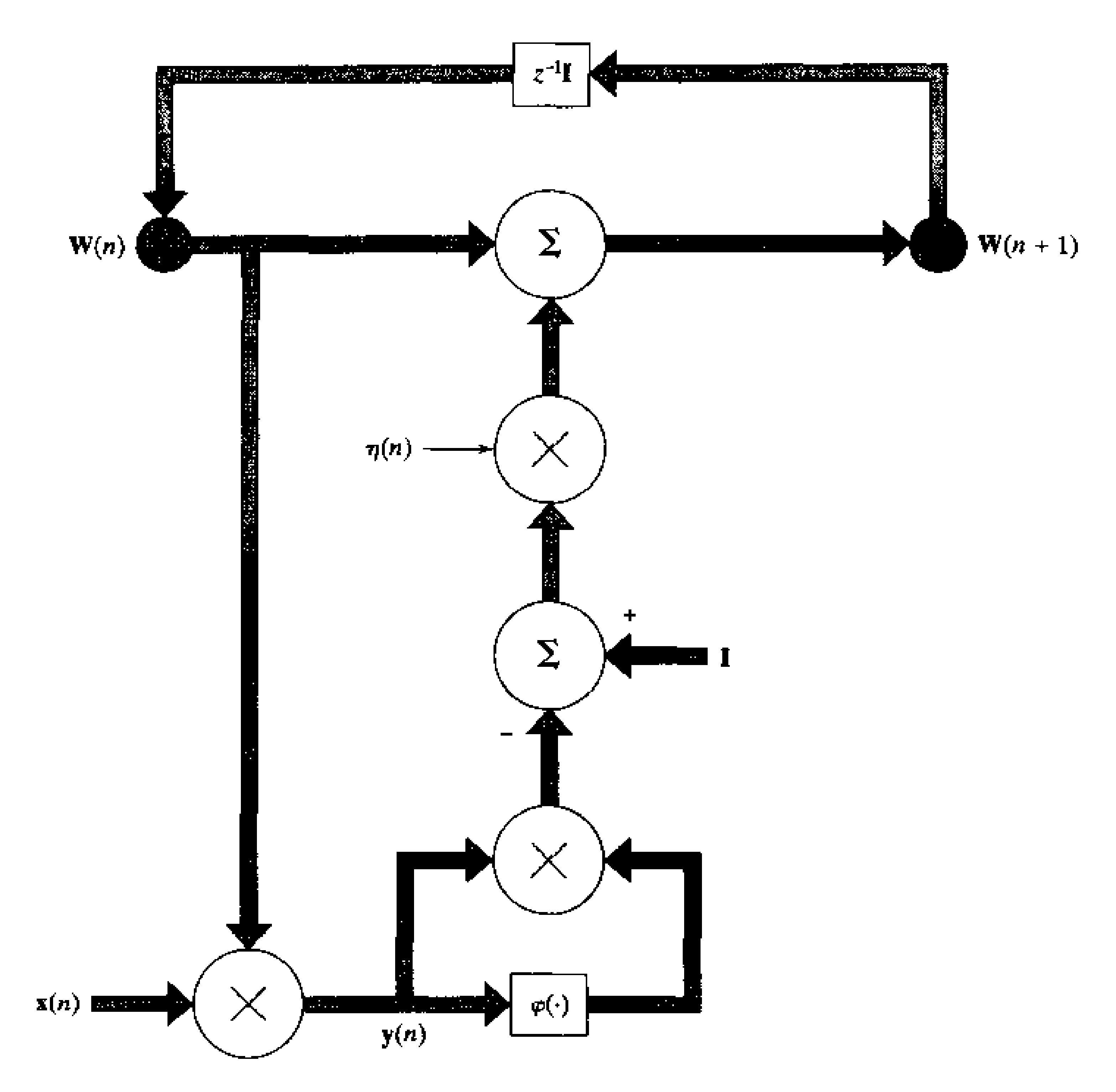


图 10-12 式(10.104)描述的盲源分离学习算法的信号流图

$$\dot{\mathbf{W}}(t) = \eta(t)[\mathbf{I} - \boldsymbol{\varphi}(\mathbf{y}(t))\mathbf{y}^T(t)]\mathbf{W}(t) \tag{10.105}$$

其中 t 表示时间，$\dot{\mathbf{W}}(t) = \partial\mathbf{W}(t)/\partial t$。对所有 t，学习率 $\eta(t)$ 总是正的。令

$$\sigma_i^2 = E[y_i^2] \tag{10.106}$$

$$k_i = E\left[\frac{\partial\varphi(y_i)}{\partial y_i}\right] \tag{10.107}$$

522

$$q_i = E\left[y_i^2\frac{\partial\varphi(y_i)}{\partial y_i}\right] \tag{10.108}$$

从而，根据 Amari et al.(1997)，对任意的激活函数 $\varphi(\cdot)$，分离解是式(10.104)的自适应性算法的稳定平衡点的充分必要条件，是对所有的 (i,j)，$i\neq j$，有

$$q_i + 1 > 0 \tag{10.109}$$

$$k_i > 0 \tag{10.110}$$

和

$$\sigma_i^2\sigma_j^2k_ik_j > 1 \tag{10.111}$$

收敛性因素

假设满足了从式(10.109)到式(10.111)的稳定性要求，我们能够对基于式(10.93)的激活

函数的式(10.104)的学习算法的收敛行为说些什么？根据 Madhuranath and Haykin(1998)所作的实验的研究报告，粗略地讲，我们可以说收敛过程有两个阶段：

- 阶段Ⅰ，经过一段时间的调整后，分离器输出的随机变量 Y_i 的方差 $\sigma_i^2(n)$ 能够达到一个相当稳定的值。在这个阶段，累积量 $\kappa_{i,3}$，$\kappa_{i,4}$ 和 $\kappa_{i,6}$ 基本保持稳定。
- 阶段Ⅱ，经过一段时间调整后累积量 $\kappa_{i,3}$，$\kappa_{i,4}$ 和 $\kappa_{i,6}$ 达到一个相当的稳定值。在这一点上，我们可以说算法已经收敛。

因此看起来分离器输出的方差和高阶累积量的估计值提供用于研究式(10.104)的学习算法收敛行为的合理程序的基础。注意这样一点是有趣的，即只在阶段Ⅱ，算法才与 Gram-Charlier 展开式一致。

10.12 计算机实验

假设图 10-9 所描述的系统包括以下三个独立的源：

$u_1(n) = 0.1\sin(400n)\cos(30n)$

$u_2(n) = 0.01\text{sgn}(\sin(500n + 9\cos(40n)))$

$u_3(n)$ = 噪声均匀分布在[-1, 1]的区间内

混合矩阵 **A** 为

$$\mathbf{A} = \begin{bmatrix} 0.56 & 0.79 & -0.37 \\ -0.75 & 0.65 & 0.86 \\ 0.17 & 0.32 & -0.48 \end{bmatrix}$$

523

信号源的波形在图 10-13 左边显示。

对于分离器，我们用式(10.104)描述更新规则的批处理形式；参见习题 10.14。选择批处理的基本原因是提高收敛性。利用以下条件实现算法：

- *初始化*：为了对算法初始化，分离矩阵 **W** 的权值用一个在[0.0,0.05]区间内均匀分布的随机数产生器选取。
- *学习率*：学习率固定在 $\eta = 0.1$。
- *信号持续时间*：在混合器的输出端产生的时间序列的采样周期为 10^{-4} 秒，包含 N = 65 000 个样本组成。

图 10-13 的右边画出经过 300 次迭代后分离器的输出端产生的信号波形。除了未知源输出的比列和置换，图 10-13 中左边的波形与右边的波形没有明显的差别。得到这里结果算法初始化权矩阵是

$$\mathbf{W}(0) = \begin{bmatrix} 0.0109 & 0.0340 & 0.0260 \\ 0.0024 & 0.0467 & 0.0415 \\ 0.0339 & 0.0192 & 0.0017 \end{bmatrix}$$

算法收敛到最后权值矩阵

524

$$\mathbf{W} = \begin{bmatrix} 0.2222 & 0.0294 & -0.6213 \\ -10.1932 & -9.8141 & -9.7259 \\ 4.1191 & -1.7879 & -6.3765 \end{bmatrix}$$

相应的矩阵积 **WA** 的值为

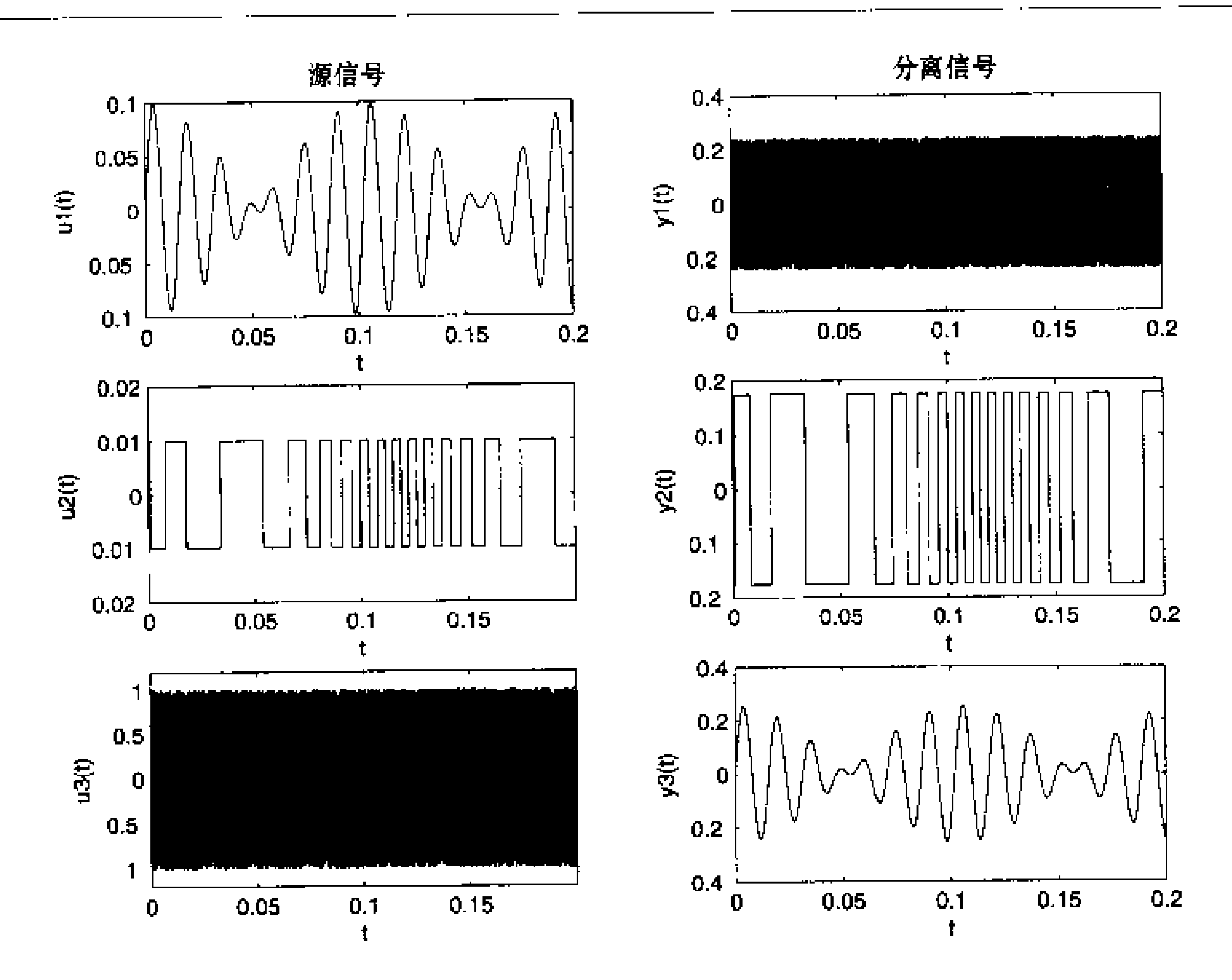


图 10-13 左边的波形：原来的源信号 右边的波形：分离后的源信号

$$
\mathbf{WA} = \begin{bmatrix} -0.0032 & -0.0041 & 0.2413 \\ -0.0010 & -17.5441 & -0.0002 \\ 2.5636 & 0.0515 & -0.0009 \end{bmatrix}
$$

重新调整矩阵积的项使得输出信号与输入信号的顺序相同，我们可写为

$$
\mathbf{WA} = \begin{bmatrix} 2.5636 & 0.0515 & -0.0009 \\ -0.0010 & -17.5441 & -0.0002 \\ -0.0032 & -0.0041 & 0.2413 \end{bmatrix}
$$

矩阵积的第一、二、三列分别对应信号的幅度调制信号、频率调制截止(clipped)信号和噪声。**WA** 中的对角元素定义图 10-13 中右边输出波形与图 10-13 左边初始信源波形之间的比例系数。

为了定量评价分离器的性能，我们可以定义一个全局拒绝指标(Amatri et al.,1996)：

$$
\mathcal{J} = \sum_{i=1}^{m}\left(\sum_{j=1}^{m}\frac{|p_{ij}|}{\max\limits_{k}|p_{ik}|}-1\right)+\sum_{j=1}^{m}\left(\sum_{i=1}^{m}\frac{|p_{ij}|}{\max\limits_{k}|p_{kj}|}-1\right)
$$

其中 $\mathbf{P}=\{p_{ij}\}=\mathbf{WA}$。性能指标$\mathcal{J}$是矩阵 **P** 对角化的量度。如果 **P** 完全对角化，则$\mathcal{J}=0$。

对于那些元素不是集中在主对角线的矩阵 **P**，其性能指数将很高。

在图 10-13 中显示的波形，$\mathcal{J}=0.0606$。

10.13 最大似然估计

前面一节所讨论的独立分量分析的方法(即最大互信息的第三种变体)只是诸多盲源分离

方法中的一种。但在信息论背景中，也仅有其他两种方法能够以无监督方式解决源分离问题：最大似然法和最大熵法。在这一节中我们讨论最大似然法。

最大似然法是一个统计估计的常用过程，具有一些良好的性质；参见第7章注释[5]。在525这个过程中，我们首先建立对数似然函数，然后根据考虑的概率模型的参数向量对它进行最优化。从第7章的讨论中，我们知道似然函数是一个给定模型中的数据集的概率密度函数，但是只是作为模型未知参数的一个函数。根据图10-9，令 $f_{\mathbf{U}}(\cdot)$ 表示随机源向量 **U** 的概率密度函数。那么在混合器输出端的观测向量 $\mathbf{X}=\mathbf{AU}$ 的概率密度函数定义为(Papoulis,1984)

$$f_{\mathbf{X}}(\mathbf{x},\mathbf{A}) = |\det(\mathbf{A})|^{-1} f_{\mathbf{U}}(\mathbf{A}^{-1}\mathbf{x}) \tag{10.112}$$

其中 $\det(\mathbf{A})$ 是混合矩阵 **A** 的行列式。令 $\mathcal{T}=\{\mathbf{x}_k\}_{k=1}^N$ 表示随机向量 **X** 的 N 次独立实现。于是可以写成

$$f_{\mathbf{X}}(\mathcal{T},\mathbf{A}) = \prod_{k=1}^{N} f_{\mathbf{X}}(\mathbf{x}_k,\mathbf{A}) \tag{10.113}$$

我们发现用归一化(除以样本数目 N)后的对数似然函数更方便，表示为

$$\begin{aligned}\frac{1}{N}\log f_{\mathbf{X}}(\mathcal{T},\mathbf{A}) &= \frac{1}{N}\sum_{k=1}^{N}\log f_{\mathbf{X}}(\mathbf{x}_k,\mathbf{A})\\ &= \frac{1}{N}\sum_{k=1}^{N}\log f_{\mathbf{U}}(\mathbf{A}^{-1}\mathbf{x}_k) - \log|\det(\mathbf{A})|\end{aligned}$$

令 $\mathbf{y}=\mathbf{A}^{-1}\mathbf{x}$ 为分离器输出端的随机向量 **Y** 的一个实现，这样可写成

$$\frac{1}{N}\log f_{\mathbf{X}}(\mathcal{T},\mathbf{A}) = \frac{1}{N}\sum_{k=1}^{N}\log f_{\mathbf{U}}(\mathbf{y}_k) - \log|\det(\mathbf{A})| \tag{10.114}$$

令 $\mathbf{A}^{-1}=\mathbf{W}$ 且 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 表示以 **W** 为参数的 **Y** 的概率密度函数。注意式(10.114)中的求和是 $\log f_{\mathbf{U}}(\mathbf{y}_k)$ 的样本平均值。从大数定律发现，当 N 趋于无穷，

$$\begin{aligned}L(\mathbf{W}) &= \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}\log f_{\mathbf{U}}(\mathbf{y}_k) + \log|\det(\mathbf{W})|\\ &= E[\log f_{\mathbf{U}}(\mathbf{y}_k)] + \log|\det(\mathbf{W})| \qquad (10.115)\\ &= \int_{-\infty}^{\infty} f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log f_{\mathbf{U}}(\mathbf{y})\,d\mathbf{y} + \log|\det(\mathbf{W})|\end{aligned}$$

以概率1成立，其中第二等式是关于 **Y** 求期望。量 $L(\mathbf{W})$ 的值是希望的对数似然函数。利用写法

$$f_{\mathbf{U}}(\mathbf{y}) = \left(\frac{f_{\mathbf{U}}(\mathbf{y})}{f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})}\right) f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$$

我们可以将 $L(\mathbf{W})$ 表示为等价形式

$$\begin{aligned}L(\mathbf{W}) &= \int_{-\infty}^{\infty} f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log\left(\frac{f_{\mathbf{U}}(\mathbf{y})}{f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})}\right)d\mathbf{y} + \int_{-\infty}^{\infty} f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\,d\mathbf{y} + \log|\det(\mathbf{W})|\\ &= -D_{f_{\mathbf{Y}}\|f_{\mathbf{U}}} - h(\mathbf{Y},\mathbf{W}) + \log|\det(\mathbf{W})| \qquad (10.116)\end{aligned}$$

526

其中 $h(\mathbf{Y},\mathbf{W})$ 是由 **W** 参数化的随机向量 **Y** 微分熵，而 $D_{f_{\mathbf{Y}}\|f_{\mathbf{U}}}$ 是 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 和 $f_{\mathbf{U}}(\mathbf{y})$ 之间的 Kullback-Leibler 散度。将式(10.76)代入式(10.116)，可将对数似然函数 $L(\mathbf{W})$ 简化成(Cardoso,1998a)

$$L(\mathbf{W}) = -D_{f_{\mathbf{Y}}\|f_{\mathbf{U}}} - h(\mathbf{X}) \tag{10.117}$$

其中 $h(\mathbf{X})$ 是分离器输入端的随机向量 **X** 的微分熵。在式(10.117)中，惟一依赖于分离器的

权值向量 **W** 的是 Kullback-Leibler 散度 $D_{f_{\mathbf{Y}} \| f_{\mathbf{U}}}$。因此从式(10.117)可以得到如下结论：最大化对数似然函数就等于最小化 Kullback-Leibler 散度 $D_{f_{\mathbf{Y}} \| f_{\mathbf{U}}}$，即使分离器的输出 **Y** 的概率分布与初始源向量 **U** 的概率分布匹配。

最大似然估计与独立分量分析之间的关系

对目前问题应用式(10.45)所描述的 Pythagoras 分解，可以将 Kullback-Leibler 散度 $D_{f_{\mathbf{Y}} \| f_{\mathbf{U}}}$ 表示为极大似然

$$D_{f_{\mathbf{Y}} \| f_{\mathbf{U}}} = D_{f_{\mathbf{Y}} \| \tilde{f}_{\mathbf{Y}}} + D_{\tilde{f}_{\mathbf{Y}} \| f_{\mathbf{U}}} \tag{10.118}$$

上式右边的第一个 Kullback-Leibler 散度 $D_{f_{\mathbf{Y}} \| \tilde{f}_{\mathbf{Y}}}$ 是表征独立分量分析方法的结构失配的度量，第二个 Kullback-Leibler 散度 $D_{\tilde{f}_{\mathbf{Y}} \| f_{\mathbf{U}}}$ 是描述初始源向量 **U** 的分布和分离器输出 **Y** 的边缘分布之间的边缘失配的度量。因此可以将用于最大似然的全局分布匹配准则表达如下(Amari, 1997; Cardoso, 1998a)：

$$\text{总失配} = \text{结构失配} + \text{边缘失配} \tag{10.119}$$

"结构失配"是指一组独立变量的一个分布的结构，而"边缘失配"是指各边缘分布之间的失配。

在理想情况下，$\mathbf{W} = \mathbf{A}^{-1}$(即完全盲源分离)所有的结构失配和边缘失配都为 0。在这种情况下，最大似然与独立分量分析产生完全相同的结果，理想情况下的两者的关系描绘在图 10-14 中(Cardoso, 1996; Amari, 1997)。在这个图中，$\mathcal{S}$ 是分离器输出端随机向量 **Y** 的所有概率 [527]

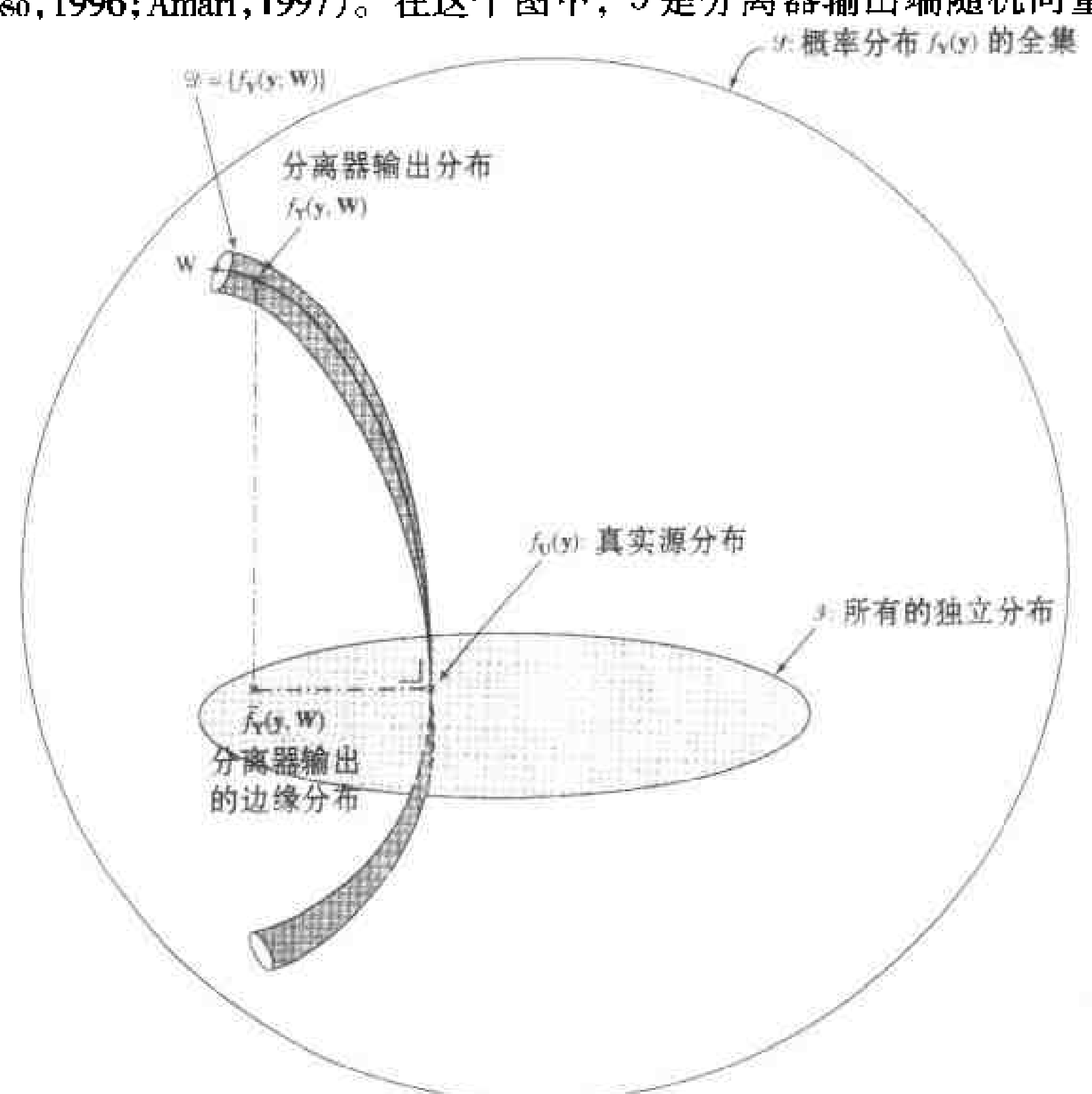


图 10-14 用于盲源分离的最大似然估计与独立分量分析之间的关系示意图
最大似然最小化 $D_{f_{\mathbf{Y}} \| f_{\mathbf{U}}}$，而独立分量分析最小化 $D_{f_{\mathbf{Y}} \| \tilde{f}_{\mathbf{Y}}}$

密度的函数$f_{\mathbf{Y}}(\mathbf{y})$的集合；$\mathscr{S}$是所有独立的概率分布的集合，也就是那些乘积形式。$\mathscr{S}$和$\mathscr{I}$都是无穷维的。集$\mathscr{D}=\{f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\}$是在分离器的输出端测量得到的概率分布的有限集。$\mathscr{D}$是$m^2$维的，其中$m$表示$\mathbf{Y}$的维数，权值向量$\mathbf{W}$是其中的一个坐标系。从图10-14中，可以清楚看出$D_{f_{\mathbf{Y}}\|\tilde{f}_{\mathbf{Y}}}$和$D_{f_{\mathbf{Y}}\|f_{\mathbf{U}}}$在$\mathbf{W}=\mathbf{A}^{-1}$时同时取得最小值。有趣的是集合$\mathscr{D}$和$\mathscr{I}$在交点处正交，该交点由真实概率密度函数$f_{\mathbf{U}}(\mathbf{y})$所定义。

对于一个基于最大似然原则的盲源分离问题算法必须包括对固有的未知源分布的估计，而这些源分布通常就是未知的。这个估计的参数正如调节分离权值矩阵$\mathbf{W}$一样是可以调节的。换句话说，我们应该进行混合矩阵和源分布(一些特征)的联合估计(Cardoso, 1997, 1998a)，这种联合估计的一种巧妙和成熟的方法已经在Pham et al.(1992,1997)中给出。

528

10.14 最大熵方法

用最大熵方法(maximum entropy method)解决盲源分离问题是由Bell and Sejnowski(1995)提出的。图10-15画出基于这种方法的系统方框图。与以前一样，分离器对观察向量$\mathbf{X}$进行操作，产生输出$\mathbf{Y}=\mathbf{WX}$，它是初始源向量$\mathbf{U}$的估计。向量$\mathbf{Y}$经过每个分量为非线性的变换$\mathbf{G}(\cdot)$变成$\mathbf{Z}$，且$\mathbf{G}(\cdot)$是一个单调可逆函数。因此，与$\mathbf{Y}$不同，对一个任意大的分离器$\mathbf{Z}$的微分熵保证都是有界的。对于给定的非线性$\mathbf{G}(\cdot)$，最大熵方法通过对$\mathbf{W}$求$h(\mathbf{Z})$的最大值，得到初始源向量$\mathbf{U}$的一个估计。根据在例10.6中导出的式(10.55)，我们看到最大熵方法与最大互信息原则是紧密相关的[17]。

非线性$\mathbf{G}$是一个对角映像，表达为

$$\mathbf{G}:\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \to \begin{bmatrix} g_1(y_1) \\ g_2(y_2) \\ \vdots \\ g_m(y_m) \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \tag{10.120}$$

我们也可以写成

$$\mathbf{Z} = \mathbf{G}(\mathbf{Y}) = \mathbf{G}(\mathbf{WAU}) \tag{10.121}$$

由于非线性$\mathbf{G}(\cdot)$是可逆的，可以将初始源向量$\mathbf{U}$利用分离器输出向量$\mathbf{Z}$表示成

$$\mathbf{U} = \mathbf{A}^{-1}\mathbf{W}^{-1}\mathbf{G}^{-1}(\mathbf{Z}) = \mathbf{\Psi}(\mathbf{Z}) \tag{10.122}$$

其中$\mathbf{G}^{-1}$是一个非线性的逆：

529

$$\mathbf{G}^{-1}:\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \to \begin{bmatrix} g_1^{-1}(z_1) \\ g_2^{-1}(z_2) \\ \vdots \\ g_m^{-1}(z_m) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \tag{10.123}$$

输出向量$\mathbf{Z}$的概率密度函数利用源向量$\mathbf{U}$的概率密度函数定义为(Papoulis, 1984)

$$f_{\mathbf{Z}}(\mathbf{z}) = \frac{f_{\mathbf{U}}(\mathbf{u})}{|\det(\mathbf{J}(\mathbf{u}))|}\Bigg|_{\mathbf{u}=\mathbf{\Psi}(\mathbf{z})} \tag{10.124}$$

其中$\det(\mathbf{J}(\mathbf{u}))$是Jacobi矩阵$\mathbf{J}(\mathbf{u})$的行列式，$\mathbf{J}(\mathbf{u})$的$ij$元素定义如下：

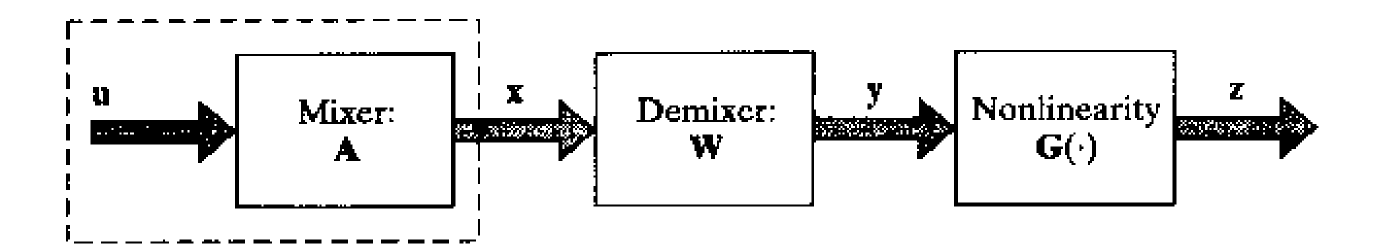


图 10-15 用于盲源分离的最大熵方法方框图
向量 **u**, **x**, **y** 和 **z** 分别是随机向量 **U**, **X**, **Y** 和 **Z** 的值

$$J_{ij} = \frac{\partial z_i}{\partial u_j} \tag{10.125}$$

所以非线性 **G** 的输出端的随机向量 **Z** 的熵为

$$h(\mathbf{Z}) = -E[\log f_{\mathbf{Z}}(\mathbf{z})] = -E\left[\log\left(\frac{f_{\mathbf{U}}(\mathbf{u})}{|\det(\mathbf{J}(\mathbf{u}))|}\right)\right]_{\mathbf{u}=\mathbf{\Psi}(\mathbf{z})}$$

$$= -D_{f_{\mathbf{U}} \| |\det \mathbf{J}|} \qquad \text{在 } \mathbf{u} = \mathbf{\Psi}(\mathbf{z}) \text{ 处求值} \tag{10.126}$$

因此可以看出求 $h(\mathbf{Z})$的最大值等价于求 $f_{\mathbf{U}}(\mathbf{u})$和由$|\det(\mathbf{J}(\mathbf{u}))|$定义的 **U** 的概率密度函数之间的 Kullback-Leibler 散度的最大值。

假设对所有的 i，随机变量 Z_i(即 **Z** 的第 i 个元素)在$[0,1]$上均匀分布。根据例 10.1，那么熵 $h(\mathbf{Z})$为 0。相应地，从式(10.126)得到

$$f_{\mathbf{U}}(\mathbf{u}) = |\det(\mathbf{J}(\mathbf{u}))| \tag{10.127}$$

在理想情况 $\mathbf{W}=\mathbf{A}^{-1}$时，这种关系化简为

$$f_{U_i}(u_i) = \left.\frac{\partial z_i}{\partial y_i}\right|_{z_i = g(u_i)} \qquad \text{对所有的 } i \tag{10.128}$$

相反，如果式(10.128)满足，则最大化 $h(\mathbf{Z})$得到 $\mathbf{W}=\mathbf{A}^{-1}$，从而盲源分离问题得到解决。

我们可以对用于盲源分离的最大熵方法得到的结果概述如下(Bell and Sejnowski, 1995)：

如图 10-15 所示，令在分离器输出的非线性由初始源分布定义为

$$z_i = g_i(y_i) = \int_{-\infty}^{z_i} f_{U_i}(u_i)\, du_i \qquad i = 1,2,\cdots,m \tag{10.129}$$

530

*最大化在非线性 **G** 输出端的随机向量 **Z** 的熵等价于 $\mathbf{W}=\mathbf{A}^{-1}$，这将产生完全的盲源分离。*

对所有的 i，在随机变量 Z_i 是区间$[0,1]$上均匀分布的条件下，最大熵方法和最大似然方法对盲源分离问题是等价的(Cardoso, 1997)。为了证明这个关系，我们首先利用微分的链式规则将式(10.125)改写为等价形式

$$J_{ij} = \sum_{k=1}^{m} \frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial x_k}\frac{\partial x_k}{\partial u_j} = \sum_{k=1}^{m} \frac{\partial z_i}{\partial y_i} w_{ik} a_{kj} \tag{10.130}$$

Jacobi 矩阵 **J** 因此可以表达为

$$\mathbf{J} = \mathbf{DWA}$$

其中 **D** 是对角矩阵

$$\mathbf{D} = \mathrm{diag}\left(\frac{\partial z_1}{\partial y_1}, \frac{\partial z_2}{\partial y_2}, \cdots, \frac{\partial z_m}{\partial y_m}\right)$$

所以

$$|\det(\mathbf{J})| = |\det(\mathbf{WA})| \prod_{i=1}^{m} \frac{\partial z_i}{\partial y_i} \tag{10.131}$$

对于由权值矩阵 **W** 和非线性函数 **G** 参数化的概率密度函数 $f_U(\mathbf{u})$，根据式(10.131)，它的估计可以形式地表示为(Roth and Baram,1996)

$$f_U(\mathbf{u} \mid \mathbf{W},\mathbf{G}) = |\det(\mathbf{WA})| \prod_{i=1}^{m} \frac{\partial g_i(y_i)}{\partial y_i} \tag{10.132}$$

从而在这种条件下，可以看出对于盲源分离最大化熵 $h(\mathbf{Z})$等价于最大化对数似然函数 $\log f_U(\mathbf{u}|\mathbf{W},\mathbf{G})$。也即是说，最大熵方法与最大似然方法是等价的。

盲源分离的学习算法

考虑到(10.126)的第二个等式，注意到由于源(信号)的分布通常是固定的，最大化熵 $h(\mathbf{Z})$要求对 **W** 求分母项 $\log|\det(\mathbf{J}(\mathbf{u}))|$的期望的最大值。我们的目标是求一个自适应算法，因此可以考虑瞬时目标函数

$$\Phi = \log|\det(\mathbf{J})| \tag{10.133}$$

将式(10.131)代入式(10.133)得到

531

$$\Phi = \log|\det(\mathbf{A})| + \log|\det(\mathbf{W})| + \sum_{i=1}^{m}\log\left(\frac{\partial z_i}{\partial y_i}\right) \tag{10.134}$$

所以对分离器的权值矩阵 **W** 求 Φ 的微分得到(见习题 10.16)

$$\frac{\partial \Phi}{\partial \mathbf{W}} = \mathbf{W}^{-T} + \sum_{i=1}^{m} \frac{\partial}{\partial \mathbf{W}}\log\left(\frac{\partial z_i}{\partial y_i}\right) \tag{10.135}$$

为了进一步处理这个公式，必须说明由分离器馈入的非线性 $\mathbf{G}(\cdot)$。这里可以使用的非线性的简单形式为 logistic 函数

$$z_i = g(y_i) = \frac{1}{1+e^{-y_i}} \qquad i = 1,2,\cdots,m \tag{10.136}$$

图 10-16 画出该函数和其反函数的图像。这个图像表明 logistic 函数满足盲源分离的单调性和可逆性的基本要求。将式(10.136)代入式(10.135)得到

532

$$\frac{\partial \Phi}{\partial \mathbf{W}} = \mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T$$

其中 **x** 是接收信号，**z** 是分离器的输出向量经非线性变化后的输出。**1** 是分量都为 1 的向量。

学习算法的目的就是最大化熵 $h(\mathbf{Z})$。因此采用最速下降法，应用于权值矩阵 **W** 的变化可表示为(Bell and Sejnowski,1995)

$$\Delta\mathbf{W} = \eta\frac{\partial \Phi}{\partial \mathbf{W}} = \eta(\mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T) \tag{10.137}$$

其中 η 是学习率参数。与独立分量分析相类似，可以利用自然梯度消除对转置权值矩阵 $\mathbf{W}^T$ 求逆的要求，这等价于对(10.137)乘以矩阵积 $\mathbf{W}^T\mathbf{W}$。这个最优调整产生权值变化所希望的公式为

$$\begin{aligned}\Delta\mathbf{W} &= \eta(\mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T)\mathbf{W}^T\mathbf{W} = \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z})(\mathbf{Wx})^T)\mathbf{W} \\ &= \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z})\mathbf{y}^T)\mathbf{W}\end{aligned} \tag{10.138}$$

其中 **y** 是分离器的输出。所以计算权值矩阵 **W** 的学习算法可以表示为

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z}(n))\mathbf{y}^T(n))\mathbf{W}(n) \tag{10.139}$$

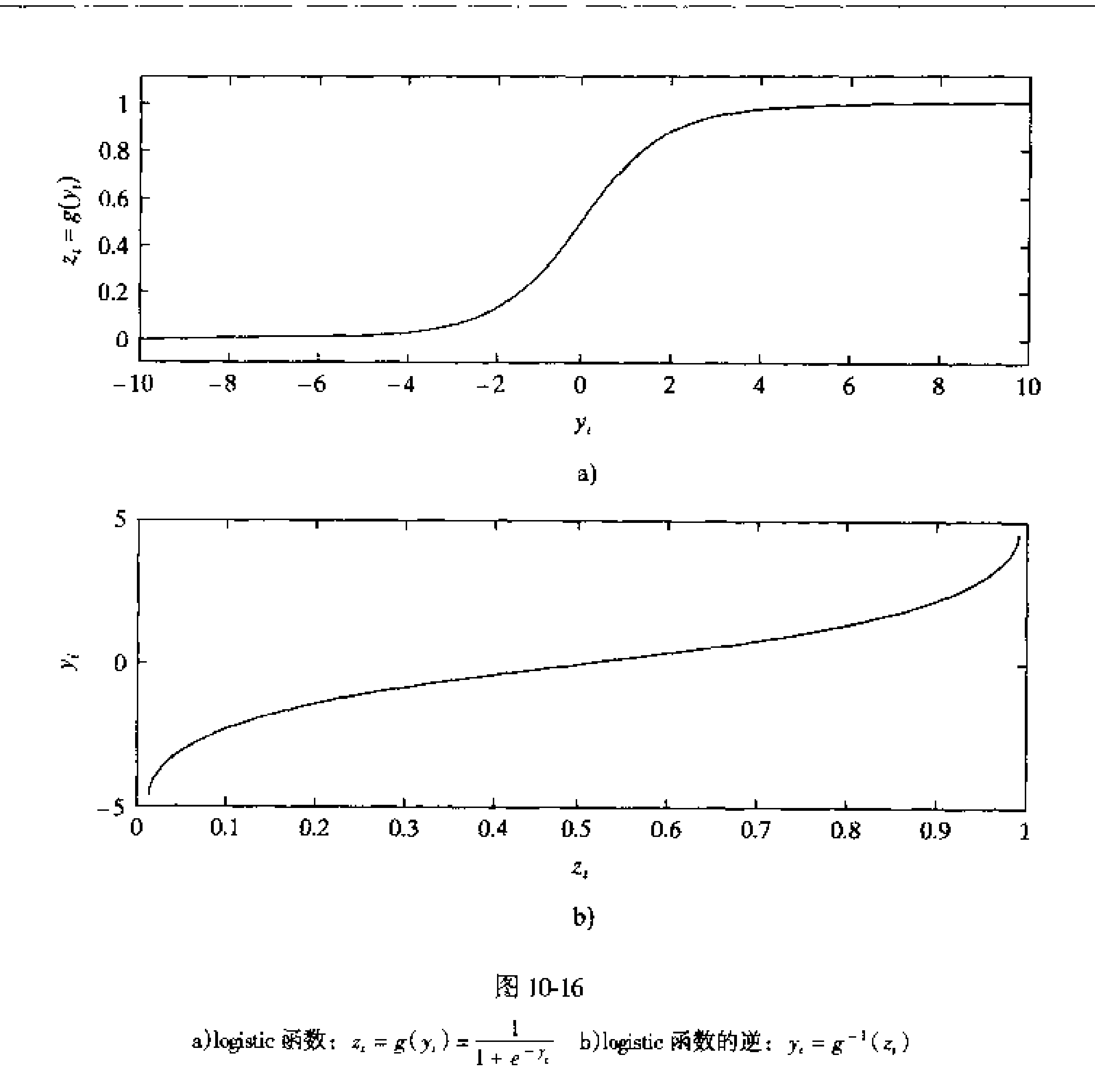


图 10-16

a) logistic 函数：$z_t = g(y_t) = \frac{1}{1+e^{-y_t}}$ b) logistic 函数的逆：$y_t = g^{-1}(z_t)$

算法的初值 $\mathbf{W}(0)$选取一组均匀分布的小数值。

理论考虑和实验观测都表明，式(10.139)的学习算法局限于分离超 Gauss 分布的源(Bell and Sejnowski, 1995)；超 Gauss 分布的定义参见注释[18]。这个局限是在图 10-15 中系统后端的非线性采用了 logistic 函数的直接结果。特别，logistic 函数对源分布加上先验知识，即一个超 Gauss 形。但是，最大熵方法限制使用 logistic 函数和最大似然方法限制某些固定先验知识并没有特别不同之处。通过修改式(10.138)的学习算法使之提供固有信源分布和混合矩阵的估计，可以将最大熵法应用到更广泛的源分布范围。这个要求同前一节讨论的最大似然法有相似的性质。

10.15 小结和讨论

在本章中，我们根据 Shannon 信息论模型，建立互信息作为自组织的基本统计工具。由于输入过程与输出过程之间的互信息有一些独特的性质，所以可以将其作为自组织学习的最优化的目标函数。实际上，一些重要的自组织原则已经出现在本章的讨论中：

- *最大互信息原则*，归功于 Linsker(1988)。这个原则的基本形式非常适合建立自组织模型和特征映射。 533
- 最大互信息的*第一种变体*，归功于 Becker and Hinton(1992)，非常适合图像处理，目

标是发现带噪声传感的输入在时间和空间上表现的相干性。

- 最大互信息的第二种变体，归功于 Ukrainec and Haykin(1992)，在对偶图像处理得到应用，对不同的两幅环境图像的相应区域中，目标是求出最大的空间差异。
- 最大互信息的第三种变形，用于独立分量分析，归功于 Comon(1994)，虽然它的基础可以追溯到 Barlow 假设(Barlow,1985,1989)。但是，在 Comon(1994)首次提出了独立分量分析的严格形式。
- 最大熵方法，归功于 Bell and Sejnowski(1995)，也是与最大互信息原则有关。最大熵与最大似然是等价的(Cardoso,1997)。

最大熵方法和独立分量分析提供两种可供选择的盲源分离的方法，并且分别有自己的性质。以最大熵原则为基础的盲源分离算法实现简单，而建立在独立分量分析基础上的相应算法在推导上更复杂，但也许有更广泛的应用。

在盲源分离中经常引用的神经生物机制是鸡尾酒会(cocktail party)现象。这种现象是指人的一种显著能力，能够在嘈杂的环境中选择性地集中并跟踪感兴趣的听觉输入。如同第2章介绍的一样，对于这个非常困难的信号处理问题的解涉及的潜在生物神经模型比图10-9所示的理想模型复杂得多。神经生物模型包括时间和空间处理形式，这是处理未知的延迟、反射和噪声所必需的。现在我们已经对标准的盲源分离问题的神经解所涉及的基本问题有了一个合理的确定理解，也许现在我们应该转移并且解决在规模上可以与鸡尾酒会相似的现实问题。

另一个值得仔细注意的公开研究领域是盲反卷积(blind deconvolution)。反卷积是一种信号处理操作，它理想地揭示线性时不变系统对输入信号所完成的卷积效果。更具体地，在通常反卷积中输出信号和系统的参数都是已知的，要求重建输入信号；在盲反卷积中，或者用更精确术语无监督反卷积，仅有输出信号是已知的并且还有可能知道源的统计信息；要求求得输入信号和系统，或两者都要求得到。很明显，盲反卷积问题是比一般的反卷积问题困难得多的信号处理任务，虽然盲反卷积在文献中得到了相当大的重视(Haykin,1994a)，与盲源分离问题一样，我们对用信息论模型的方法来解决盲反卷积问题的研究还处在初级阶段(Douglas and Haykin,1997)。而且，在诸如移动通信通道之类的不友好信道中盲平衡(blind equalization)的代价有效解，和鸡尾酒会问题的解一样是一个挑战性的问题。

534

总的来说，盲自适应问题，在源分离或反卷积的环境下，要达到监督学习那样的成熟发展阶段还需要很长的一段路要走。

注释和参考文献

[1] 想进一步了解信息论，请参考 Cover and Thomas(1991)的书；也可参考 Gray(1990)的书。如果想参考信息论发展的论文集(包括1948年 Shannon 的经典论文)，可见 Slepian(1973)。Shannon 的论文经过一些小的改动被重版在 Shannon and Weaver(1949)的书和 Sloane and Wyner(1993)的书中。

想对在神经处理中的信息论原则作一个简短的回顾，可参考 Atick(1992)。想从生物的角度来理解信息论方法，可参考 Yockey(1992)。

[2] 不要把 Linsker 的自组织最大互信息原则与决策系统的信息内容保存规则相混淆，后者是在第7章简要讨论的经验规则。

[3] 信息论与感知之间关系的文献综述可以参考 Linsker(1990c)和 Atick(1992)。

[4] 息论理论中的术语"熵"的名字是从热力学中的熵衍生来的；热力学中的熵由

$$H = -k_B \sum_{\alpha} p_\alpha \log p_\alpha$$

定义(见第11章)，其中 k_B 是 Boltzman 常数，p_α 是系统处于状态 α 的概率。除了系数 k_B 之外热力学中的熵 H 的公式与我们在(10.8)给出得熵的定义在数学形式上是一致的。

[5] Shore and Johnson(1980)中证明在如下意义下最大熵原则是正确的：
以约束形式给出先验知识，在满足这些约束的分布中根据"相容性公理"(consistency axioms)能够选择惟一的分布；这个惟一的分布由最大化熵定义。
相容性公理有四部分：
Ⅰ. 惟一性：结果必须是惟一的。
Ⅱ. 不变性：坐标的选择应当不影响结果。
Ⅲ. 系统独立性：无论用不同密度或用联合密度来解释独立系统的独立信息应该是无关紧要的。
Ⅳ. 子集独立性：无论用分离的条件密度或用完整的系统密度来处理独立的系统状态子集应该是无关紧要的。
Shore and Johnson(1980)证明相对熵或 Kullback-Leibler 散度同样满足相容性公理。

[6] 关于 Lagrange 乘子法的讨论，参考 Dorny(1975)的书。

535

[7] 项 $I(X;Y)$，最初 Shannon(1948)用来指信息传输率。今天，而我们用来指随机变量 X 和 Y 之间的互信息。

[8] 为了证明式(10.45)的分解，可以处理如下。由定义我们有

$$\begin{aligned} D_{f_{\mathbf{X}} \| f_{\mathbf{U}}} &= \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{f_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) d\mathbf{x} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{f_{\mathbf{X}}(\mathbf{x})}{\tilde{f}_{\mathbf{X}}(\mathbf{x})} \cdot \frac{\tilde{f}_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) d\mathbf{x} \\ &= \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{f_{\mathbf{X}}(\mathbf{x})}{\tilde{f}_{\mathbf{X}}(\mathbf{x})}\right) d\mathbf{x} + \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{f}_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) d\mathbf{x} \qquad (1) \\ &= D_{f_{\mathbf{X}} \| \tilde{f}_{\mathbf{X}}} + \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{f}_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) d\mathbf{x} \end{aligned}$$

从 $\tilde{f}_{\mathbf{X}}(\mathbf{x})$ 和 $f_{\mathbf{U}}(\mathbf{x})$ 的定义得到

$$\log\left(\frac{\tilde{f}_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) = \log\left(\frac{\prod_{i=1}^{m} \tilde{f}_{X_i}(x_i)}{\prod_{i=1}^{m} f_{U_i}(x_i)}\right) = \sum_{i=1}^{m} \log\left(\frac{\tilde{f}_{X_i}(x_i)}{f_{U_i}(x_i)}\right)$$

用 B 表示式(1)最后等式中的积分，可以写成

$$B = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{f}_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{U}}(\mathbf{x})}\right) d\mathbf{x} = \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\prod_{i=1}^{m} \tilde{f}_{X_i}(x_i)}{\prod_{i=1}^{m} f_{U_i}(x_i)}\right) d\mathbf{x}$$

$$= \sum_{i=1}^{m} \int_{-\infty}^{\infty} \left(\log\left(\frac{\tilde{f}_{X_i}(x_i)}{f_{U_i}(x_i)}\right) \int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x}^{(i)}\right) dx_i = \sum_{i=1}^{m} \int_{-\infty}^{\infty} \log\left(\frac{\tilde{f}_{X_i}(x_i)}{f_{U_i}(x_i)}\right) \tilde{f}_{X_i}(x_i)\, dx_i \qquad (2)$$

在上面最后的等式中用了式(10.39)的定义。式(2)的积分是 Kullback-Leibler 散度，

$D_{\bar{f}_{X_i} \| f_{U_i}}, i=1,2,\cdots,m$。为了把 B 表达成最后的形式，注意函数 $\bar{f}_{X_j}(x_j)$ 下面的面积是 1，因此可写为

$$B = \sum_{i=1}^{m} \int_{-\infty}^{\infty} \prod_{j=1}^{m} \bar{f}_{X_j}(x_j) \left(\log\left(\frac{\bar{f}_{X_i}(x_i)}{f_{U_i}(x_i)} \right) dx_i \right) d\mathbf{x}^{(i)}$$

$$= \int_{-\infty}^{\infty} \bar{f}_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\prod_{i=1}^{m} \bar{f}_{X_i}(x_i)}{\prod_{i=1}^{m} f_{U_i}(x_i)} \right) d\mathbf{x} = D_{\bar{f}_{\mathbf{X}} \| f_{\mathbf{U}}} \tag{3}$$

536

其中在第一个等式利用定义了 $d\mathbf{x} = dx_i d\mathbf{x}^{(i)}$，如同在 10.5 节描述的一样。因此，将(3)代入(1)，我们得到期望的分解：

$$D_{f_{\mathbf{X}} \| f_{\mathbf{U}}} = D_{f_{\mathbf{X}} \| \bar{f}_{\mathbf{X}}} + D_{\bar{f}_{\mathbf{X}} \| f_{\mathbf{U}}}$$

[9] Nadal and Parga(1994，1997)也讨论最大互信息和冗余减少之间的关系，得到同样的结果：神经系统的输入向量和输出向量之间的互信息的最大化也就导致数据减少。Haft and van Hemmen(1998)讨论视网膜的最大互信息过滤器的实现情况。结果表明，对于像视网膜这样的感觉系统所产生的内部环境表示，冗余性对获得噪声鲁棒性是最根本的。

[10] Becker and Hinton(1992)用字母 I_{max} 表示最大互信息的第一种变体。

[11] 在 Uttley(1970)中考虑负信息通路，通过最优化通路中输入信号与输出信号之间的互信息的负值。结果表明，这样的系统在调整期间适宜变成输入信号集中更常发生的模式的判别器。这种模型被称作"informon"，它与最大互信息的第二种变体有松散关系。

[12] 在 Ukrainec and Haykin(1996)中描述的系统包括一个后探测处理器，它利用关于反射器沿水道的水陆边界位置的先验知识。模糊处理器结合初始探测性能和基于视觉的边缘检测器的输出以便有效地去除错误警报，从而得到系统性能的进一步提高。

[13] 盲源分离问题可追溯至 Hérault，Jutten and Ans(1985)的启蒙性文章。对盲源分离问题的历史记载，参考 Nadal and Parga(1997)；这篇文章也强调问题的神经生物侧面。强调固有信号处理原则的盲源分离问题的深刻综述可以参考 Cardoso(1998a)。

[14] **概率密度函数逼近**

(a)Gram-Charlier 展开式

令随机变量 Y 的概率密度为 $f_Y(y)$，$\varphi_Y(\omega)$ 是它的特征函数。根据定义我们有

$$\varphi_Y(\omega) = \int_{-\infty}^{\infty} f_Y(y) e^{j\omega y} dy \tag{1}$$

其中 $j=\sqrt{-1}$，ω 是实数。总的说来，除了指数形式的符号改变外，特征函数 $\varphi_Y(\omega)$ 是概率密度函数 $f_Y(y)$ 的傅里叶变换。一般意义上，特征函数 $\varphi_Y(\omega)$ 是一个复数，它的实部和虚部对所有 ω 是有限的。如果随机变量 Y 的 k 阶矩存在，则 $\varphi_Y(\omega)$ 可以在 $\omega=0$ 处展开成幂级数

$$\varphi_Y(\omega) = 1 + \sum_{k=1}^{\infty} \frac{(j\omega)^k}{k!} m_k \tag{2}$$

其中 m_k 是 Y 的 k 阶矩，定义为

537

$$m_k = E[Y^k] = \int_{-\infty}^{\infty} y^k f_Y(y) dy \tag{3}$$

式(2)的导出只是简单地将式(1)中的指数函数用其展开式代替，交换求和与积分的顺序，并利用式(3)。如果特征函数能展开为式(2)的形式，我们可以将 $\varphi_Y(\omega)$ 的对数展开为(Wilks, 1962)

$$\log\varphi_Y(\omega) = \sum_{n=1}^{\infty} \frac{\kappa_n}{n!}(j\omega)^n \tag{4}$$

其中 κ_n 是随机变量 Y 的 n 阶累积量或半不变量(semi-invariant)。式(4)是由 $\log\varphi_Y(\omega)$ 在 $\omega=0$ 处的关于 $j\omega$ 的 Taylor 展开式得到。

为了简化问题，我们作如下两个假设：

1. 随机变量 Y 的均值为 0，即 $\mu = 0$。
2. Y 的方差是归一化的，即 $\sigma^2 = 1$。

相应地，有 $\kappa_1 = 0$，$\kappa_2 = 1$，而式(4)的展开式变成

$$\log\varphi_Y(\omega) = \frac{1}{2}(j\omega)^2 + \sum_{n=3}^{\infty} \frac{\kappa_n}{n!}(j\omega)^n \tag{5}$$

现在令

$$r(\omega) = \sum_{n=3}^{\infty} \frac{\kappa_n}{n!}(j\omega)^n$$

可以将式(5)重写成

$$\log\varphi_Y(\omega) = \frac{1}{2}(j\omega)^2 + r(\omega)$$

也就是说，特征函数可以表示成两个指数函数相乘的形式：

$$\varphi_Y(\omega) = \exp\left(-\frac{\omega^2}{2}\right) \cdot \exp(r(\omega)) \tag{6}$$

将 $\exp(r(\omega))$ 展开成幂级数得到

$$\exp(r(\omega)) = 1 + \sum_{l=1}^{\infty} \frac{r^l(\omega)}{l!} \tag{7}$$

538

将式(7)代入式(6)，在双重求和中按 $(j\omega)$ 的幂进行整理，得到 $\varphi_Y(\omega)$ 展开式的新系数：

$$c_1 = 0, c_2 = 0, c_3 = \frac{\kappa_3}{6}, c_4 = \frac{\kappa_4}{24}, c_5 = \frac{\kappa_5}{120}$$

$$c_6 = \frac{1}{720}(\kappa_6 + 10\kappa_3^2), c_7 = \frac{1}{5040}(\kappa_7 + 35\kappa_4\kappa_3), c_8 = \frac{1}{40320}(\kappa_8 + 56\kappa_5\kappa_3 + 35\kappa_4^2)$$

等等。现在可以用 $\varphi_Y(\omega)$ 的逆傅里叶变换求概率密度函数的展开式 $f_Y(y)$。特别可以写成

$$f_Y(y) = \alpha(y)\left(1 + \sum_{k=3}^{\infty} c_k H_k(y)\right) \tag{8}$$

其中的 $\alpha(y)$ 是零均值和单位方差的归一化 Gauss 随机变量的概率密度函数：

$$\alpha(y) = \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \tag{9}$$

展开式(8)就称为由 Gauss 函数和它的导数表示的概率密度函数的 Gram-Charlier 级数(Stuart and Ord, 1994)。这种形式的展开式具有直观性好处。特别地，如果随机变量 Y 是由一些独立的同分布的随机变量的和，那么当变量的数目趋于无穷时，根据中心极

限定理，Y 趋于 Gauss 分布。Gram-Charlier 级数展开式的第一项确实是 Gauss 的，这意味着它的表示随着变量的数目增加，序列后面的项的和趋近于零。

式(8)中的 Hermite 多项式 $H_k(y)$ 通过 $\alpha(y)$ 的 k 阶导数定义为

$$\alpha^{(k)}(y) = (-1)^k \alpha(y) H_k(y) \tag{10}$$

下面是一些典型的 Hermite 多项式：

$$H_0(y) = 1, H_1(y) = y, H_2(y) = y^2 - 1,$$
$$H_3(y) = y^3 - 3y, H_4(y) = y^4 - 6y^2 + 3,$$
$$H_5(y) = y^5 - 10y^3 + 15y, H_6(y) = y^6 - 15y^4 + 45y^2 - 15$$

这些多项式的递推关系为

$$H_{k+1}(y) = yH_k(y) - kH_{k-1}(y) \tag{11}$$

Hermite 多项式一个特别有用的性质是 $H_k(y)$ 和 Gauss 函数 $\alpha(y)$ 的 m 次导数是双正交的，表示为

539

$$\int_{-\infty}^{\infty} H_k(y)\alpha^{(m)}(y)dy = (-1)^m m!\delta_{km}, \qquad (k,m) = 0,1,\cdots \tag{12}$$

其中 δ_{km} 是 Kronecker 符号，当 $k = m$ 为它为 1，其他情况为 0。

重要的是注意到项的自然顺序对 Gram-Charlier 级数并不是最好的。相反，应按下面列出的圆括号内的项分组(Helstrom,1968)：

$$k = (0),(3),(4,6),(5,7,9) \tag{13}$$

这些分组的元素经常是同一数量级的。例如我们保留 $k = 4$ 的项，则我们也应当包括 $k = 6$。

(b)Edgeworth 展开式

与前面一样，令 $\alpha(y)$ 是一个归一化为零均值和方差为 1 的随机变量的概率密度函数。随机变量 Y 的概率密度函数对 Gauss 逼近 $\alpha(y)$ 的 Edgeworth 展开式为(Comon,1994; Stuart and Ord,1994)

$$\begin{aligned}\frac{f_Y(y)}{\alpha(y)} = 1 &+ \frac{\kappa_3}{3!}H_3(y) + \frac{\kappa_4}{4!}H_4(y) + \frac{10\kappa_3^2}{6!}H_6(y) + \frac{\kappa_5}{5!}H_5(y) \\ &+ \frac{35\kappa_3\kappa_4}{7!}H_7(y) + \frac{280\kappa_3^3}{9!}H_9(y) + \frac{\kappa_6}{6!}H_6(y) + \frac{56\kappa_3\kappa_5}{8!}H_8(y) \\ &+ \frac{35\kappa_4^2}{8!}H_8(y) + \frac{2100\kappa_3^2\kappa_4}{10!}H_{10}(y) + \frac{15400\kappa_3^4}{12!}H_{12}(y) + \cdots\end{aligned} \tag{14}$$

其中 κ_i 表示标准化后的标量随机变量 Y 的 i 阶累积量，H_i 表示 i 阶 Hermite 多项式。式(14)叫做 Edgeworth 级数。

Edgeworth 展开式的关键特征是系数为一致递减的。另一方面，式(8)中的 Gram-Charlier 展开式从数值误差来看并不一致趋于 0；也就是说，一般地，没有哪一项可以被忽略。也就是这个原因，才要按照式(13)的分组来截断 Gram-Charlier 展开式。

[15] 用 $\nabla^* D = (\nabla D)\mathbf{W}^T\mathbf{W}$ 代替通常梯度 ∇D 解决盲源分离问题的思想在 Cardoso and Laheld (1996)中有详细的介绍。这里 $\nabla^* D$ 称为相对梯度，这个梯度与自然梯度是相同的。自然梯度是从信息几何的观点来定义的(Amari,1998; Amari et al.1996)。类似的算法早些时候在 Cichocki and Moszczyński(1992)和 Cichocki et al.(1994)中有描述。

[16] 例如，在 n 维黎曼空间中，向量 $\mathbf{a}$ 的平方范数定义为

$$\|\mathbf{a}\|^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} g_{ij}a_i a_j$$

其中 g_{ij} 是黎曼空间坐标 x_1，x_2，…，x_n 的函数，$g_{ij}=g_{ji}$，表达式右边总是正的。这个表达式是欧几里德平方范数公式

$$\|\mathbf{a}\|^2 = \sum_{i=1}^{n} a_i^2$$

540

的推广。关于黎曼空间结构的讨论，参考 Amari(1987)和 Murray and Rice(1993)。

[17] 根据式(10.55)定义熵 $H(\mathbf{Y})$ 和互信息 $I(\mathbf{Y};\mathbf{X})$ 之间的关系，Bell and Sejnowski(1995)把他们的盲源分离的方法称为最大互信息原则。但是更好的术语是“最大熵方法”，因为它涉及熵 $h(\mathbf{Z})$ 的最大化，其中 $\mathbf{Z}=\mathbf{G}(\mathbf{Y})$。归功于 Bell 和 Sejnowski 的盲源分离的最大熵方法不应与归功于 Burg(1975)的谱分析的最大熵方法(MEM)相混淆。

[18] 随机变量 X 被称为亚高斯(sub-Gauss)的(Benveniste et al.,1987)，如果

- 它是均匀分布的。
- 它的概率密度函数 $f_X(x)$ 可以表示成 $\exp(-g(x))$ 的形式，其中 $g(x)$ 可能除了原点外为可微的偶函数，并且 $g(x)$ 和 $g'(x)/x$ 在区间 $(0,\infty)$ 是严格递增的。例如，可能取 $g(x)=|x|^\beta$，$\beta>2$。

但是，如果 $g'(x)/x$ 在 $(0,\infty)$ 是递减的，而其他的性质都满足，则随机变量 X 就叫超高斯(super-Gauss)的(Benveniste et al.,1987)，例如 $g(x)=|x|^\beta$，$\beta<2$。

有时(也许有些滥用的方式)使用随机变量的峭度(kurtosis)符号作为亚高斯或超高斯的指标。随机变量 X 的峭度定义为

$$K_4(x) = \frac{E[X^4]}{(E[X^2])^2} - 3$$

在此基础上，根据峭度 $K_4(x)$ 为负或为正，随机变量 X 分别称为亚高斯或超高斯的。

习题

最大熵原则

10.1 随机变量 X 的支撑集(也就是取非零的值域)定义为 $[a,b]$；没有别的限制加在 X 上。该随机变量的最大熵分布是什么？证明你的结论。

互信息

10.2 推导 10.4 节描述的随机变量 X 和 Y 的互信息 $I(X;Y)$ 的特性。

10.3 假设输入随机向量 $\mathbf{X}$ 由初始分量 $\mathbf{X}_1$ 和背景分量 $\mathbf{X}_2$ 组成，定义

$$Y_i = \mathbf{a}_i^T\mathbf{X}_1,\ Z_i = \mathbf{b}_i^T\mathbf{X}_2$$

试问 Y_i 和 Z_i 之间的互信息和 $\mathbf{X}_1$ 和 $\mathbf{X}_2$ 之间的互信息有何关系？假设向量 $\mathbf{X}$ 的概率模型是多元 Gauss 分布

$$f_{\mathbf{X}}(\mathbf{x}) = \frac{1}{(2\pi)^{m/2}(\det\boldsymbol{\Sigma})^{1/2}}\exp((\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))$$

541

其中 $\boldsymbol{\mu}$ 是 $\mathbf{X}$ 的均值，$\boldsymbol{\Sigma}$ 是它的协方差矩阵。

10.4 在这个习题中我们探索用相对熵或 Kullback-Leibler 散度来推导多层感知机

(Hopfield, 1987b; Baum and Wilczek, 1998)的监督学习算法。更确切地说，考虑一个由一个输入层、一个隐藏层和一个输出层构成的多层感知机，假设实例或样本 α 呈现给输入，输出层神经元 k 的输出解释为概率：

$$y_{k|\alpha} = p_{k|\alpha}$$

相应的，令 $q_{k|\alpha}$表示当输入是 α 时，假设 k 为真的条件概率的实际值，该多层感知机的相对熵定义为

$$D_{p\|q} = \sum_{\alpha} p_{\alpha} \sum_{k} \left(q_{k|\alpha} \log\left(\frac{q_{k|\alpha}}{p_{k|\alpha}} \right) + (1 - q_{k|\alpha}) \log\left(\frac{1 - q_{k|\alpha}}{1 - p_{k|\alpha}} \right) \right)$$

其中 p_{α} 是一个出现 α 情况的先验概率。

以 $D_{p\|q}$为最优化的代价函数，推导一个多层感知机的学习算法。

最大互信息原则

10.5 假设有两个通道。它们的输出分别用随机变量 X 和 Y 表示，要求使 X，Y 之间的互信息达到最大。证明只要满足以下条件则就可以达到要求：

(a)出现 X 的概率和出现 Y 的概率分别是 0.5。

(b)X，Y 的联合概率密度函数集中在概率空间的一个小区域内。

10.6 考虑图 10-17 中的噪声模型，两个神经网络的输入端都为 m 个源节点。输入由 X_1，X_2，…，X_m 表示，相应的输出结果用 Y_1，Y_2 表示。你可以作如下假设：

- 网络输出端的加性噪声分量 N_1，N_2 是 Gauss 分布，具有零均值和共同方差 σ_N^2，并且互不相关。
- 每个噪声源与输入信号无关。
- 输出信号 Y_1，Y_2 都是 0 均值的 Gauss 分布。

(a)求输出向量 $\mathbf{Y} = [Y_1, Y_2]$与输入向量 $\mathbf{X} = [X_1, X_2, \cdots, X_m]^T$ 之间的互信息。

(b)利用(a)中导出的结果，检测在以下情况下冗余/相异性是如何折中的(Linsher, 1998a)：

542

(i)噪声的方差很大，表示为 σ_N^2 相对于 Y_1，Y_2 很大。

(ii)噪声的方差很小，表示为 σ_N^2 相对于 Y_1，Y_2 很小。

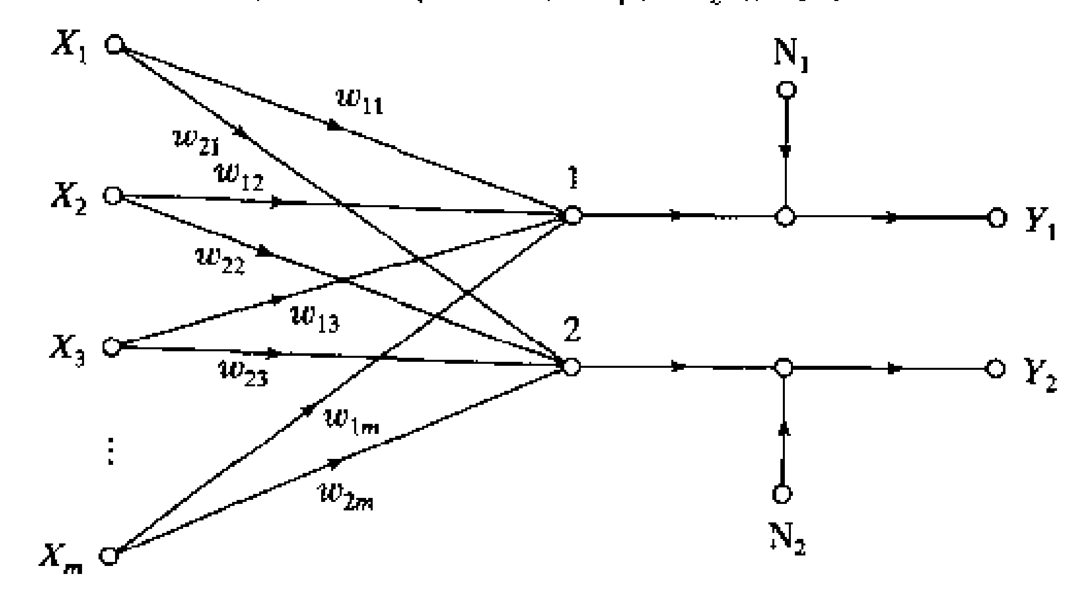


图 10-17

10.7 在 10.9 节中所描述的最大互信息原则的变体(Becker and Hinton, 1992)中，目标是根据噪声神经系统的输入向量 $\mathbf{X}_a$，$\mathbf{X}_b$ 求输出 Y_a，Y_b 之间的互信息 $I(Y_a; Y_b)$的最大值。

在 Becker and Hinton (1992)讨论的另一个方法中，一个不同的目标是求输出 Y_a 和 Y_b 的平均值与它们固有的共同信号分量 S 之间的互信息 $I\left(\frac{Y_a+Y_b}{2};S\right)$ 的最大值。

利用式(10.59)和式(10.60)中描述的噪声模型，完成下列任务：

(a)证明

$$I\left(\frac{Y_a+Y_b}{2};S\right)=\log\left(\frac{\mathrm{var}[Y_a+Y_b]}{\mathrm{var}[\mathrm{N}_a+\mathrm{N}_b]}\right)$$

其中 N_a，N_b 是 Y_a，Y_b 相应的噪声分量。

(b)用信号加噪声与噪声的比来解释此互信息。

独立分量分析

10.8 给出主分量分析(在第 8 章讨论)与独立分量分析(在本章讨论)的详细比较。

10.9 独立分量分析可以用作检测和分类之前近似数据分析的预处理步骤(Comon, 1994)。讨论能在这种应用中加以利用的独立分量分析的性质。

10.10 Darmois *定理*陈述只有当各个独立变量是 Gauss 分布的，其和才是 Gauss 分布的(Darmois,1993)。用独立分量分析证明这个定理。

10.11 在实际的应用中，一个独立分量分析算法实现只能得到“尽可能统计独立”。比较用该算法解盲源分离问题得到的解与利用去相关方法得到的解的差异。假设观察向量的协方差矩阵为非奇异的。

10.12 参考图 10-9 描述的系统，证明分离器的输出 $\mathbf{Y}$ 的任何两个分量的互信息最小化与参数化的概率密度函数 $f_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 和相应的析因分布 $\tilde{f}_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 之间的 Kullback-Leibler 散度的最小化等价。

10.13 在式(10.104)中描述的盲源分离问题的自适应算法有两个重要的性质：(1)等变化性，(2)权值矩阵 $\mathbf{W}$ 保持非奇异。性质(1)在 10.11 节后面部分有详细的介绍。在本习题中我们考查第二个性质。 [543]

假设用于开始(10.104)算法的初始值 $\mathbf{W}(0)$ 满足的条件 $|\det(\mathbf{W}(0))|\neq 0$，证明对所有的 n 有 $|\det(\mathbf{W}(n))|\neq 0$。这是保证 $\mathbf{W}(n)$ 对所有的 n 是非奇异的充分必要条件。

10.14 在这个问题中，我们讨论式(10.104)所描述的盲源分离算法的批处理公式。特别地，我们写成

$$\Delta\mathbf{W}=\eta\left(\mathbf{I}-\frac{1}{N}\mathbf{\Phi}(\mathbf{Y})\mathbf{Y}^T\right)\mathbf{W}$$

其中

$$\mathbf{Y}=\begin{bmatrix} y_1(1) & y_1(2) & \cdots & y_1(N)\\ y_2(1) & y_2(2) & \cdots & y_2(N)\\ \vdots & \vdots & & \vdots\\ y_m(1) & y_m(2) & \cdots & y_m(N)\end{bmatrix}$$

$$\mathbf{\Phi}(\mathbf{Y})=\begin{bmatrix} \varphi(y_1(1)) & \varphi(y_1(2)) & \cdots & \varphi(y_1(N))\\ \varphi(y_2(1)) & \varphi(y_2(2)) & \cdots & \varphi(y_2(N))\\ \vdots & \vdots & & \vdots\\ \varphi(y_m(1)) & \varphi(y_m(2)) & \cdots & \varphi(y_m(N))\end{bmatrix}$$

其中 N 是可用数据点的数目。证明上式描述的权值矩阵 $\mathbf{W}$ 的调整 $\Delta\mathbf{W}$ 的公式成立。

最大熵方法

10.15 考虑图 10-15，我们得到

$$\mathbf{Y} = \mathbf{W}\mathbf{X}$$

其中

$$\mathbf{Y} = [Y_1, Y_2, \cdots, Y_m]^T$$

$$\mathbf{X} = [X_1, X_2, \cdots, X_m]^T$$

$\mathbf{W}$ 是一个 $m \times m$ 的权值矩阵。令

$$\mathbf{Z} = [Z_1, Z_2, \cdots, Z_m]^T$$

其中

$$Z_k = \varphi(Y_k), \quad k = 1, 2, \cdots, m$$

(a)证明 $\mathbf{Z}$ 的联合熵与 Kullback-Leibler 散度 $D_{f\|\tilde{f}}$之间的关系为

$$h(\mathbf{Z}) = -D_{f\|\tilde{f}} - D_{\tilde{f}\|q}$$

其中 $D_{\tilde{f}\|q}$是下面两个量的 Kullback-Leibler 散度：(a)统计独立的(即析因式的)输出向量组 $\mathbf{Y}$ 的概率密度函数，(b)由 $\prod_{i=1}^{m} q(y_i)$定义概率密度函数。

(b)对所有的 i，当 $q(y_i)$与初始源输出 U_i 的概率密度函数相等时，$h(\mathbf{Z})$的公式该如何修改?

10.16 (a)从式(10.134)开始，推导式(10.135)的结果。

544

(b)用式(10.136)中的 logistic 函数，证明使用式(10.135)将产生由式(10.137)给出的公式。

第11章　植根于统计力学的随机机器和它们的逼近

11.1　简介

作为我们无监督(自组织)学习系统的最后一种类别，我们以统计力学作为我们思想的出发点。统计力学的主题围绕对大系统宏观平衡态性质的形式化研究，而系统的每个基本元素服从力学的微观定律。统计力学的主要目标是从微观元素如原子和电子的运动推导出宏观物体的热力学性质(Landau and Lifshitz,1980;Parisi,1988)。这里面对的自由度数量是巨大的，这样只有利用统计的方法进行研究。正如 Shannon 的信息论一样，在统计力学的研究中熵的概念起着关键的作用：系统越有序或者它的概率分布越集中，则熵越小。同样我们可以说，系统越无序或它的概率分布越均匀，则熵越大。在 1975 年，Jaynes 证明了熵不仅可以像前一章所述的那样作为构造统计推理的出发点，而且可以作为产生统计力学研究基础的 Gibbs 分布的出发点。

利用统计力学作为研究神经网络基础的兴趣可以追溯到 Cragg and Temperley(1954)以及 Cowan(1968)的早期工作。Boltzmann 机(Hinton&Sejnowski,1983,1986;Ackley et al.,1985)也许是第一个由统计力学导出的多层学习机。机器命名的原因是为了表明神经网络自己的动力学行为和 Boltzmann 初始的统计热力学的形式的等价性。基本上说，Boltzmann 机可以对给定数据集的固有概率分布进行建模，这样在诸如模式完备和模式分类等任务中所使用的条件分布就可以导出来了。令人遗憾的是 Boltzmann 机的学习过程是令人难以忍受地慢，这一缺点导致对 Boltzmann 机的修改和产生了新的随机机器。以上这些问题构成了本章的大部分题材。

本章的组织

本章被组织成三部分。第一部分由 11.2 节至 11.6 节所组成。11.2 节给出统计力学的简要评述，在 11.3 节中回顾一类特殊类型的随机过程，即 Markov 链，它是在研究统计力学中常会遇到的。11.4 节、11.5 节和 11.6 节描述三种随机模拟技巧：Metropolis 算法、模拟退火和 Gibbs 抽样。

本章的第二部分由 11.7 至 11.9 节组成，讨论三类随机机器。11.7 节描述 Boltzmann 机。11.8 节描述 sigmoid 信度网络。11.9 节描述另一类新的称为 Helmholtz 机的随机机器。

本章的最后一部分由 11.10 节至 11.13 节组成，讨论随机机器的基于统计力学中的平均场理论的逼近。11.10 节讨论在一般意义下的平均场理论。11.11 节讨论 Boltzmann 机的平均场理论，随后的 11.12 节讨论对 sigmoid 信度网络平均场理论更原则性的处理方法。11.13 节描述一种对模拟退火的逼近，即确定退火。

本章最后在 11.14 节中给出最终的评论。

11.2 统计力学

考虑具有许多自由度的物理系统，它可以驻留在大量可能状态中的任何一个。例如，用 p_i 表示状态 i 发生的概率，具有如下性质：

$$p_i \geqslant 0 \quad 任给\ i \tag{11.1}$$

和

$$\sum_i p_i = 1 \tag{11.2}$$

用 E_i 表示系统在状态 i 时的能量，统计热力学告诉我们，当系统和它周围的环境处于热平衡时，一个基本的结果是状态 i 发生的概率如下：

546

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{k_B T}\right) \tag{11.3}$$

其中 T 为开尔文绝对温度，k_B 为 Boltzmann 常数，Z 为与状态无关的常数。1 开尔文度相当于摄氏 -273 度，$k_B = 1.38\times 10^{-23}$ 焦耳/开。

式(11.2)定义概率规范化的条件。将这个条件加入到式(11.3)我们有

$$Z = \sum_i \exp\left(-\frac{E_i}{k_B T}\right) \tag{11.4}$$

规范化量 Z 称为*状态和*或者*剖分函数*(通常用符号 Z 是因为这项的德文名字为 Zustadsumme)。式(11.3)的概率分布称为*典型分布*或 Gibbs *分布*[1]；指数因子 $\exp(-E_i/k_B T)$ 称为 Boltzmann 因子。

对 Gibbs 分布以下两点值得注意：

1. 能量低的状态比能量高的状态发生的概率高；
2. 随着温度 T 降低，概率集中在低能状态的一个更小的子集上。

在神经网络的领域内，就我们主要关心的内容而言，温度 T 可以被视为一种*伪温度*，它控制表示一个神经元突触噪声的热波动。它的精确标度因而无关重要。相应地，我们可以置常数 k_B 为单位而重新度量之，因此重新定义概率 p_i 和剖分函数 Z 如下：

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{T}\right) \tag{11.5}$$

和

$$Z = \sum_i \exp\left(-\frac{E_i}{T}\right) \tag{11.6}$$

今后我们处理统计力学就在这两个定义基础上进行，其中 T 简单称为*系统温度*。从式(11.5)我们注意到 $-\log p_i$ 可以被看作在单位温度下“能量”的一种度量。

自由能量和熵

一个物理系统的 Helmholtz *自由能量*记为 F，由剖分函数 Z 定义如下：

$$F = -T\log Z \tag{11.7}$$

系统的平均能量定义为

$$\langle E \rangle = \sum_i p_i E_i \tag{11.8}$$

其中 $\langle\cdot\rangle$ 表示总体平均运算。因此，利用式(11.5)至式(11.8)，可以看出平均能量和自由能量之差为

$$< E > - F = - T \sum_i p_i \log p_i \qquad (11.9)$$ 547

式(11.9)右边的量忽略温度 T，称为系统的熵，表示为

$$H = - \sum_i p_i \log p_i \qquad (11.10)$$

因此我们重可以写式(11.9)为

$$< E > - F = TH$$

的形式或等价的

$$F = < E > - TH \qquad (11.11)$$

考虑两个系统 A 和 A' 彼此热接触。假设系统 A 比系统 A' 更小，这样 A' 可以看作具有恒温 T 的热储藏器。两个系统的总熵趋向于依照关系式(Reif,1967)

$$\Delta H + \Delta H' \geqslant 0$$

增加，其中 ΔH 和 $\Delta H'$ 分别表示系统 A 和 A' 熵的改变量。根据式(11.11)，这个关系的涵义是指自由能量逐渐降低至平衡态时变为最小。由统计力学我们发现此时它的概率分布为 Gibbs 分布。因而我们有一个重要的原则称为最小自由能量原则，它可以陈述如下(Landau and Lifshitz,1980;Parisi 1988)：

随机系统变元的自由能量的最小值可在热平衡时达到，此时系统服从 Gibbs 分布。

自然偏爱具有最小自由能量的物理系统。

11.3 Markov 链

考虑一个由多个随机变量组成系统，其演化可由一个随机过程 $\{X_n, n = 1,2,\cdots\}$ 描述。随机变量 X_n 在时刻 n 取值 x_n 称为系统在 n 时刻的状态。随机变量所有可能的值构成的空间称为系统的状态空间。如果随机过程 $\{X_n, n = 1,2,\cdots\}$ 的构造使得 X_{n+1} 的条件概率分布仅依靠于 X_n 的值而与其他以前的值无关，我们所这个过程为 Markov 链(Feller,1950;Ash,1965)。更准确地说，我们有

$$P(X_{n+1} = x_{n+1} \mid X_n = x_n, \cdots, X_1 = x_1) = P(X_{n+1} = x_{n+1} \mid X_n = x_n) \qquad (11.12)$$ 548

这称之为 Markov 特性。换句话说：

如果系统在 $n+1$ 时刻出现状态 x_{n+1} 的概率仅依赖于系统在 n 时刻出现状态 x_n 的概率，则随机变量序列 X_1，X_2，$\cdots$，X_n，X_{n+1} 成为 Markov 链。

因此我们可以将 Markov 链看作产生模型，它由一些状态和可能的状态转移链接而成。每时刻访问一个特定的状态，模型输出一个该状态相关的符号。

转移概率

在 Markov 链中，从一个状态到另一个状态的转移是随机的，但输出符号却是确定的。令

$$p_{ij} = P(X_{n+1} = j \mid X_n = i) \qquad (11.13)$$

表示在 n 时刻状态 i 转移到 $n+1$ 时刻状态 j 的转移概率。既然 p_{ij} 为条件概率，所有的转移概率必须满足两个条件：

$$p_{ij} \geq 0 \quad 对所有(i,j) \tag{11.14}$$

$$\sum_{j} p_{ij} = 1 \quad 对所有\ i \tag{11.15}$$

我们将假定转移概率是固定的，不随时间改变；也就是说，式(11.13)对所有时间 n 成立。在这种情况下，Markov 链称为关于时间是齐次的。

如果系统具有有限数目的可能状态，例如 K 个状态，则转移概率构成一个 $K \times K$ 的矩阵

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & & \vdots \\ p_{K1} & p_{K2} & \cdots & p_{KK} \end{bmatrix} \tag{11.16}$$

它的元素满足式(11.14)和式(11.15)所叙的条件，而后一条件就是 $\mathbf{P}$ 的每行的和为 1。这种类型的矩阵称为随机矩阵。任何随机矩阵可以作为转移概率矩阵。

由式(11.13)定义的一步转移概率可以推广到经过固定的步数从一个状态转移到另一个状态。令 $p_{ij}^{(m)}$ 表示从状态 i 到状态 j 的 m 步转移概率：

$$p_{ij}^{(m)} = P(X_{n+m} = x_j \mid X_n = x_i), m = 1,2,\cdots \tag{11.17}$$

我们可以把 $p_{ij}^{(m)}$ 看作系统从状态 i 转移到状态 j 经历的所有中间状态 k 的和。特别地，$p_{ij}^{(m+1)}$ 可由 $p_{ij}^{(m)}$ 递推而得：

549

$$p_{ij}^{(m+1)} = \sum_{k} p_{ik}^{(m)} p_{kj}, m = 1,2,\cdots \tag{11.18}$$

而

$$p_{ik}^{(1)} = p_{ik}$$

式(11.18)可以推广如下：

$$p_{ij}^{(m+n)} = \sum_{k} p_{ik}^{(m)} p_{kj}^{(n)}, \quad (m,n) = 1,2,\cdots, \tag{11.19}$$

这是 Chapman-Kolmogorov 恒等式的特殊情形(Feller,1950)。

当链上的一个状态仅能在 d 的整数倍时刻上重新出现，我们说该状态有周期 d。一个 Markov 链称为非周期的，如果它的所有状态仅有周期 1。

常返性

假设一个 Markov 链从状态 i 开始，它以概率 1 返回状态 i，则称状态 i 为常返的；也就是说

$$f_i = P(总是返回状态\ i) = 1$$

若概率 $f_i < 1$，则称状态 i 为瞬态(Leon-Garcia,1994)。

如果 Markov 链从一个常返态开始，则该状态在时间上将无穷次重现。如果从一个瞬态开始，它将只能有限次重现。这可以作如下解释。我们可以把状态 i 重新发生看作一个成功概率为 f_i 的 Bernoulli 试验。它返回的次数为具有均值$(1-f_i^{-1})$的几何随机变量。若 $f_i < 1$，这意味着有无穷次成功的次数为零。因此一个瞬态确实在有限次返回后不再发生(Leon-Garcia,1994)。

如果一个 Markov 链有某些瞬态和常返状态，则该过程最终只会在常返态之间移动。

不可约 Markov 链

一个 Markov 链上的状态 j 称为从状态 i 可达的，如果从状态 i 到 j 存在有限步具有正概率的转移。如果状态 i 和状态 j 之间互为可达的，则该 Markov 链的状态 i 和状态 j 称为彼此相通的。这种相通可写作 $i \leftrightarrow j$。很明显，如果状态 i 与状态 j 相通，且状态 j 与状态 k 相通，即 $i \leftrightarrow j$ 和 $j \leftrightarrow k$，则状态 i 和状态 k 相通，即 $i \leftrightarrow k$。

如果一个 Markov 链的两个状态相通，它被说成是属于同一类的。一般情况下，一个 Markov 链的状态组成一个或多个不相通的类。但是，如果所有状态组成一个类，则称该 Markov 链为不可分的或不可约的。换句话说，一个不可约的 Markov 链从任一个状态开始，可以以正的概率达到任何别的状态。可约链在大多数的应用领域无实际价值。相应地我们限制我们的注意仅在不可约的链。 550

考虑一个不可约的 Markov 链，在时刻 $n=0$ 时开始于常返态 i。令 $T_i(k)$ 表示第 $k-1$ 次和第 k 次返回状态 i 之间的时间间隔。状态 i 的平均常返时间定义为 $T_i(k)$ 关于 k 的期望值。状态 i 的稳态概率，记为 π_i，等于平均常返时间 $E[T_i(k)]$ 的倒数，即由下式表示：

$$\pi_i = \frac{1}{E[T_i(k)]}$$

若 $E[T_i(k)] < \infty$，也就是 $\pi_i > 0$，状态 i 称为正常返的。若 $E[T_i(k)] = \infty$，也就是 $\pi_i = 0$，状态 i 称为零常返的。$\pi_i = 0$ 意味着 Markov 链最终达到的状态再返回状态 i 是不可能的。正常返和零常返是不同类的性质，这意味着同时具有正常返和零常返的 Markov 链是可约的。

遍历 Markov 链

大体上说，遍历性意味着我们可以用时间的平均替代总体平均。对一个 Markov 链来说，遍历性意味着链处于状态 i 的时间长度和稳态概率 π_i 相对应，这可以说明如下。k 次返回后花费在状态 i 的时间比 $\nu_i(k)$ 定义为

$$\nu_i(k) = \frac{k}{\sum_{l=1}^{k} T_i(l)}$$

返回时间 $T_i(l)$ 构成一列独立的和同分布的随机变量，因为由定义，每次返回的时间都是和以前返回的时间统计独立的。更进一步，对常返态 i，链返回状态 i 无穷次。因此当返回次数 k 逼近无穷大时，大数定律表明，花费在状态 i 的时间比例趋近稳态概率，表示为

$$\lim_{k \to \infty} \nu_i(k) = \pi_i, i = 1,2,\cdots,K \tag{11.20}$$

Markov 链为遍历的一个充分但不必要的条件是它为不可约的且非周期的。

收敛于平衡分布

考虑一个遍历的 Markov 链，相应的转移矩阵为 $\mathbf{P}$。令行向量 $\boldsymbol{\pi}^{(n-1)}$ 表示链在 $n-1$ 时刻的状态分布向量；$\boldsymbol{\pi}^{(n-1)}$ 的第 j 个分量为在时刻 $n-1$ 时链处于状态 x_j 的概率。在 n 时刻状态分布向量可以定义为

551

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(n-1)}\mathbf{P} \tag{11.21}$$

由(11.21)迭代得到

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(n-1)}\mathbf{P} = \boldsymbol{\pi}^{(n-2)}\mathbf{P}^2 = \boldsymbol{\pi}^{(n-3)}\mathbf{P}^3 = \cdots$$

并且最后可以写成

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(0)}\mathbf{P}^n \tag{11.22}$$

其中 $\boldsymbol{\pi}^{(0)}$ 为状态分布向量的初始值。也就是说，Markov 链在时刻 n 状态分布向量为初始状态分布向量 $\boldsymbol{\pi}^{(0)}$ 和随机矩阵 $\mathbf{P}$ 的 n 次方的乘积。

令 $p_{ij}^{(n)}$ 表示 $\mathbf{P}^n$ 的第 ij 个元素。假设随时间 n 趋向无穷大时，$p_{ij}^{(n)}$ 趋于与 i 无关的 π_j，其中 π_j 为状态 j 的稳态概率。相应地，对于大的 n，矩阵 $\mathbf{P}^n$ 逼近于有相等行的方阵形式，可表示为

$$\lim_{n\to\infty}\mathbf{P}^n = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_K \\ \pi_1 & \pi_2 & \cdots & \pi_K \\ \vdots & \vdots & & \vdots \\ \pi_1 & \pi_2 & \cdots & \pi_K \end{bmatrix} = \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\pi} \\ \vdots \\ \boldsymbol{\pi} \end{bmatrix} \tag{11.23}$$

其中 $\boldsymbol{\pi}$ 是行向量由 π_1，π_2，…，π_K 构成。从而我们由(11.22)发现(经过一系列调整)

$$\left[\sum_{j=1}^{K}\pi_j^{(0)} - 1\right]\boldsymbol{\pi} = \mathbf{0}$$

因为由定义 $\sum_{j=1}^{K}\pi_j^{(0)} = 1$，初始分布的独立向量 $\boldsymbol{\pi}$ 满足这个条件。

现在我们可以叙述关于 Markov 链的遍历定理如下(Feller, 1950; Ash, 1965)：

设一个遍历且不可约的 Markov 链具有状态 x_1，x_2，…，x_K 和随机矩阵 $\mathbf{P}=\{p_{ij}\}$。那么，该链有惟一的平稳分布，可以由任一初始态收敛到它；也就是说，存在惟一一组数 $\{\pi_j\}_{j=1}^K$ 使得

1. $\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j$ 对于所有 i (11.24)
2. $\pi_j > 0$ 对于所有 j (11.25)
3. $\sum_{j=1}^{K}\pi_j = 1$ (11.26)

552

4. $\pi_j = \sum_{i=1}^{K}\pi_i p_{ij}$ 对于 $j = 1,2,\cdots,K$ (11.27)

相反，假定一个 Markov 链为非周期不可约的，存在 $\{\pi_j\}_{j=1}^K$ 满足式(11.25)至(11.27)，那么该链是遍历的，π_j 由式(11.24)给出，状态 j 的平均常返时间为 $1/\pi_j$。

概率分布函数 $\{\pi_j\}_{j=1}^K$ 称为不变分布或平稳分布。这样命名是因为它一旦建立，将永远保持。根据遍历定理，我们可以断言：

- 从任意初始分布开始，一个 Markov 链的转移概率将收敛于一个平稳分布，只要这个平稳分布存在。
- 遍历的 Markov 链的平稳分布独立于它的初始分布。

例 11.1 考虑一个 Markov 链，其状态转移图由图 11-1 描绘，它有两个状态 x_1 和 x_2。链的随机矩阵为

$$\mathbf{P} = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

它满足式(11.4)和式(11.5)的条件。假设初始条件是 $\pi^{(0)} = \begin{bmatrix} \frac{1}{6} & \frac{5}{6} \end{bmatrix}$，由式(11.21)我们发现在时刻 $n = 1$ 状态分布向量为

$$\pi^{(1)} = \pi^{(0)}\mathbf{P} = \begin{bmatrix} \frac{1}{6} & \frac{5}{6} \end{bmatrix} \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{11}{24} & \frac{13}{24} \end{bmatrix}$$

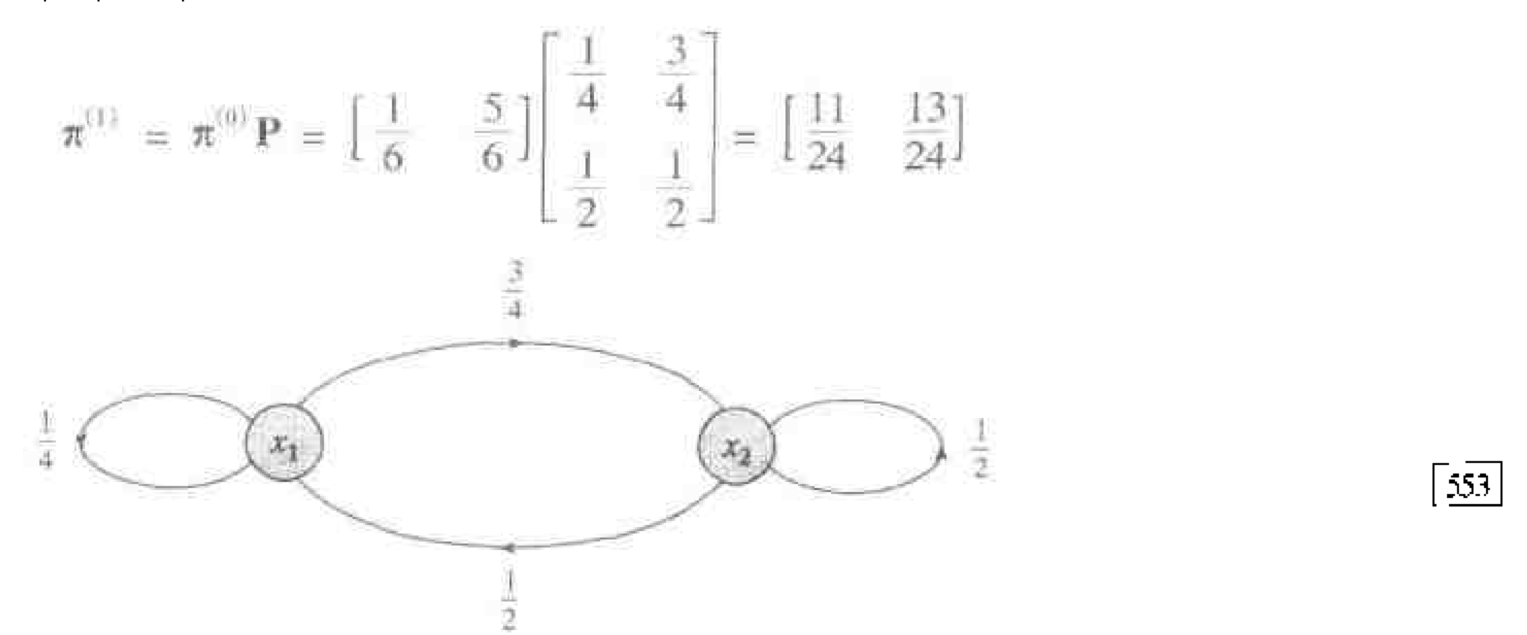


图 11-1 例 11.1 的 Markov 链的状态转移图

[553]

升高随机矩阵 $\mathbf{P}$ 的幂次为 $n = 2, 3, 4$，我们有

$$\mathbf{P}^2 = \begin{bmatrix} 0.4375 & 0.5625 \\ 0.3750 & 0.6250 \end{bmatrix}$$

$$\mathbf{P}^3 = \begin{bmatrix} 0.4001 & 0.5999 \\ 0.3999 & 0.6001 \end{bmatrix}$$

$$\mathbf{P}^4 = \begin{bmatrix} 0.4000 & 0.6000 \\ 0.4000 & 0.6000 \end{bmatrix}$$

因此 $\pi_1 = 0.4000$ 和 $\pi_2 = 0.6000$。在这个例子中，平稳分布的收敛基本上在 $n = 4$ 次迭代就完成了。由于 π_1 和 π_2 都大于零，两个状态都是正常返的，并且链为不可约的。同时注意它是非周期的，这是因为使 $(\mathbf{P}^{(n)})_{jj} > 0$ 的所有正整数 n 的最大公约数等于1。因此我们得出结论，图 11-1 所示的 Markov 链是遍历的。 ■

例 11.2 考虑随机矩阵具有某些零元素的 Markov 链，如

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 \\ \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} & 0 \end{bmatrix}$$

该链的状态转移图由图 11-2 描绘。

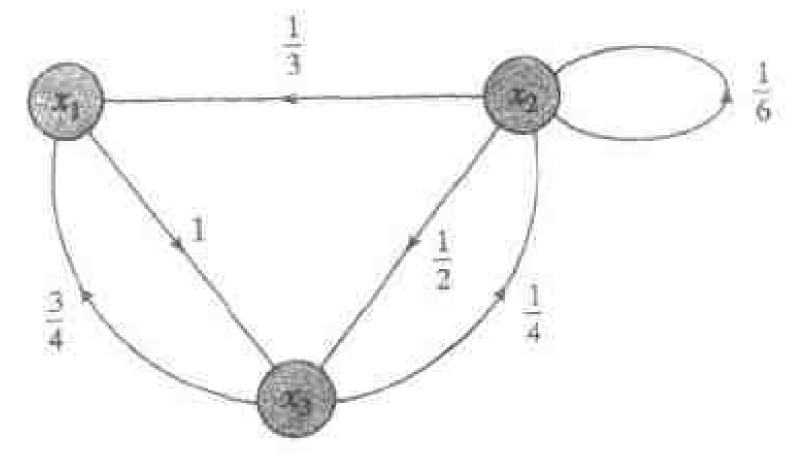


图 11-2 例 11.2 的 Markov 链的状态转移图

应用式(11.27)得到下列联立方程组：

$$\pi_1 = \frac{1}{3}\pi_2 + \frac{3}{4}\pi_3, \pi_2 = \frac{1}{6}\pi_2 + \frac{1}{4}\pi_3, \pi_3 = \pi_1 + \frac{1}{2}\pi_2$$

[554]

解关于 π_1，π_2 和 π_3 的方程组，我们得

$$\pi_1 = 0.3953, \quad \pi_2 = 0.1395, \quad \pi_3 = 0.4652$$

这个给定的 Markov 链是遍历的，它的平稳分布由 π_1，π_2 和 π_3 定义。■

状态分类

在所述材料的基础上，我们可以对状态所属的类进行小结，如图 11-3 所示(Feller,1950; Leon-Garlin,1994)。这个图还包括状态相关的长期行为。

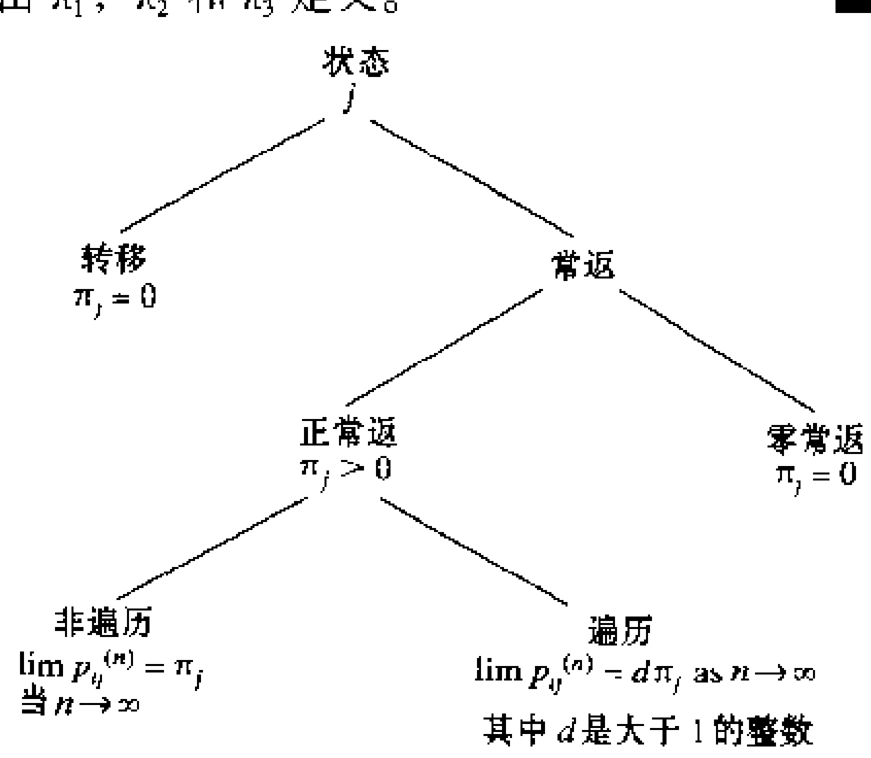


图 11-3 Markov 链的状态分类及它们相应的长期行为

细节平衡原则

式(11.25)和式(11.26)仅仅强调数值 π_j 为概率。式(11.27)是关键的，因为不可约的 Markov 链必须满足它，从而也就有平稳分布存在。式(11.27)可以认为是一阶反应动力学中的细节平衡原则的重新陈述。细节平衡原则表明，在热平衡中任何转移的发生率等于对应的逆转移的发生率，可表达为(Reif,1965)

555

$$\pi_i p_{ij} = \pi_j p_{ji} \tag{11.28}$$

为了导出式(11.27)的关系，我们可以对等式的左边进行求和如下：

$$\sum_{i=1}^{K} \pi_i p_{ij} = \sum_{i=1}^{K} \left(\frac{\pi_i}{\pi_j} p_{ij} \right) \pi_j = \sum_{i=1}^{K} (p_{ji}) \pi_j = \pi_j$$

在等式的第二行中我们应用了细节平衡原则，在最后一行利用了一个 Markov 链的转移概率满足的条件(参看式(11.15)，其中交换了 i 和 j 的作用)：

$$\sum_{i=1}^{K} p_{ji} = 1 \quad 对所有\ j$$

注意细节平衡原则意味着分布 $\{\pi_j\}$ 是一个平稳分布。

11.4 Metropolis 算法

至此我们弄清了 Markov 链的构成，我们将应用它构成一个模拟物理系统演化到热平衡的随机算法。这个算法称为 Metropolis 算法(Metropolis et al.,1953)。它是 Monte Carlo 方法的一种修改，在早期的科学计算中 Monte Carlo 方法是对大量原子在给定温度下的平衡态的随机模拟。

假设随机变量 X_n 表示任一 Markov 链在时刻 n 的状态为 x_i。我们随机生成新的状态 x_j，它表示另一个随机变量 Y_n 的一次实现。假设生成这个新状态满足对称条件：

$$P(Y_n = x_j \mid X_n = x_i) = P(Y_n = x_i \mid X_n = x_j)$$

令 ΔE 表示系统从状态 $X_n = x_i$ 到状态 $Y_n = x_j$ 所产生的能量差。如果能量差为负，则这次转移导致一个较低能量状态且这次转移被接受。这个新状态也就接受作为算法下步的起点，即我们令 $X_{n+1} = Y_n$。反之如果能量差 ΔE 为正，这时算法以概率方式进行处理。首先，我们

选择一个在单位区间[0,1]上均匀分布的随机数 ξ。如果 $\xi < \exp(-\Delta E/T)$，其中 T 为操作温度，转移被接受且置 $X_{n+1} = Y_n$。否则，转移被拒绝，置 $X_{n+1} = X_n$；即旧的配置被算法的下一步重新利用。

转移概率的选择

对任意 Markov 链，设它有先验转移概率，记为 τ_{ij}，它满足三个条件： 556

1. 非负性：$\tau_{ij} \geq 0$ 对所有(i,j)

2. 归一化：$\sum_j \tau_{ij} = 1$ 对所有 i

3. 对称性：$\tau_{ij} = \tau_{ji}$ 对所有(i,j)

令 π_i 表示 Markov 链在状态 x_i，$i = 1, 2, \cdots, K$ 的平稳态概率。因而我们可以利用已定义的对称的 τ_{ij} 和概率分布比 π_j/π_i 来构成期望的转移概率(Beckerman,1997)：

$$p_{ij} = \begin{cases} \tau_{ij}\left(\dfrac{\pi_j}{\pi_i}\right) & \text{对于}\dfrac{\pi_j}{\pi_i} < 1 \\ \tau_{ij} & \text{对于}\dfrac{\pi_j}{\pi_i} \geq 1 \end{cases} \tag{11.29}$$

为了确保转移概率归一化为单位 1，我们引入无转移概率的附加定义：

$$p_{ii} = \tau_{ii} + \sum_{j \neq i} \tau_{ij}\left(1 - \frac{\pi_j}{\pi_i}\right) = 1 - \sum_{j \neq i} \alpha_{ij}\tau_{ij} \tag{11.30}$$

其中 α_{ij} 是移动概率，定义为

$$\alpha_{ij} = \min\left(1, \frac{\pi_j}{\pi_i}\right) \tag{11.31}$$

惟一尚需解决的要求是怎样选择比值 π_j/π_i。为满足这个要求，我们选择概率分布使得所得的 Markov 链收敛到一个 Gibbs 分布，表示为

$$\pi_j = \frac{1}{Z}\exp\left(-\frac{E_j}{T}\right)$$

这时概率分布比 π_j/π_i 取简单形式

$$\frac{\pi_j}{\pi_i} = \exp\left(-\frac{\Delta E}{T}\right) \tag{11.32}$$

其中

$$\Delta E = E_j - E_i \tag{11.33}$$

557

利用概率分布比可以排除对剖分函数 Z 的依赖。

根据构造，转移概率是非负的且归整化为单位 1，如式(11.14)和式(11.15)的要求。进一步，它们满足由式(11.28)所定义的细节平衡原则。这个定律对热平衡是一个充分条件。为了说明满足细节平衡原则，我们给出下列的考虑：

情况 1： $\Delta E < 0$。假设从状态 π_i 转移到状态 π_j，能量变化 ΔE 为负。从式(11.32)我们发现 $\pi_j/\pi_i > 1$，所以利用式(11.29)得到

$$\pi_i p_{ij} = \pi_i \tau_{ij} = \pi_i \tau_{ji}$$

和

$$\pi_j p_{ji} = \pi_j\left(\frac{\pi_i}{\pi_j}\tau_{ji}\right) = \pi_i \tau_{ji}$$

因此当 $\Delta E<0$ 时细节平衡原则满足。

情况2： $\Delta E>0$。假设从状态 x_i 到状态 x_j 的能量变化 ΔE 为正，这时我们发现 $(\pi_j/\pi_i)<1$，利用式(11.29)得到

$$\pi_i p_{ij} = \pi_i\left(\frac{\pi_j}{\pi_i}\tau_{ij}\right) = \pi_j\tau_{ij} = \pi_j\tau_{ji}$$

和

$$\pi_j p_{ji} = \pi_j\tau_{ji}$$

这里又看出细节平衡原则得到满足。

为了完整起见，我们需要指出由 τ_{ij} 表示的先验转移概率的使用。这些转移概率事实上是 Metropolis 算法中的随机步的概率模型。由前面给出的算法描述，我们回忆随机步后面是随机决策。因此可以得出结论，利用通过由先验转移概率 τ_{ij} 在式(11.29)和式(11.30)定义的转移概率 p_{ij} 和平稳概率分布 π_j 对 Metropolis 算法来说确实是正确的选择。

值得注意的是由 Metropolis 算法产生的平稳分布并不惟一决定 Markov 链。平稳态时的 Gibbs 分布也可以利用其他更新规则而不是 Metropolis 算法的 Monte Carlo 规则产生；例如利用由 Ackley et al.(1986)提出的 Boltzmann 学习规则产生；这个规则将在11.7节中讨论。

11.5 模拟退火

考虑寻找一个低能量系统的问题，其状态由一个 Markov 链排序。由式(11.11)观察到当温度 T 趋近于零，系统的自由能量 F 趋近平均能量 $<E>$。由 $F\to<E>$，我们观察到由自由能量最小化原则，该 Markov 链的平稳分布即 Gibbs 分布，当 $T\to 0$ 时坍塌到平均能量 $<E>$ 的全局极小点。换句话说，序列中的低能状态在低温时受到更强的支持。这些观察促使我们提出问题：为什么不简单地应用 Metropolis 算法产生大量的代表该随机系统在很低温度下的构形(Configuration)？我们不提倡使用这种策略是因为在很低温度下 Markov 链到热平衡的收敛速度特别慢。而提高计算效率更好的方法是在较高温度运行随机系统，这时达到平衡态的收敛相当快，接着随温度的精细下降保持系统的平衡态。也就是，我们使用两个相关成分的组合：

[558]

- 一个决定温度下降速度的调度表
- 一个算法——如 Metropolis 算法——迭代求解每个调度表给出的新的温度下的平衡分布，这时利用前面温度时的最终状态作为新温度时的起始点。

我们刚才提到的两步格式是被广泛使用的以*模拟退火*[2]著称的随机松弛技术的精华(Kirkpatrick et al.,1983)。这个技术的名字是类比物理/化学中的退火过程得到的，在物理/化学的退火过程中，我们从高温度开始退火过程，接着慢慢降低温度同时保持热平衡。

模拟退火最初的目标是寻找刻划复杂大系统的代价函数的全局极小点[3]。正是因为如此，它提供一个求解非凸最优化问题的有力工具，这由下面的简单想法所导致：

当优化一个非常复杂的大系统(即具有许多自由度的系统)时不要求总是下降而是试图要求大部分时间在下降。

模拟退火在两方面和传统的迭代优化算法不同：

- 算法不会陷入局部最小，因为当系统在非零温度上运行时脱离局部最小总是可能的；
- 模拟退火是自适应的，在高温时看见系统的终态的大致轮廓，而它的具体细节在低温度时才呈现出来。

退火进度表

如前面提到的，模拟退火过程的基础是 Metropolis 算法，其间温度 T 慢慢下降。也就是说，温度 T 起调节参数的作用。假定温度下降得不比对数更快，则模拟退火过程将收敛于一个具有最小能量的构形。遗憾的是这种退火进度太慢了——慢得不切实用。实际上，我们必须求诸于算法的渐进收敛的有限时间逼近，这种逼近所付出的代价是算法不再以概率 1 保证找到全局最小点。然而算法的逼近结果在许多实际应用上能产生近似最优解。 559

为了实现模拟退火算法的有限时间逼近，我们必须设定一系列控制算法收敛的参数，这些参数组合成所谓的*退火进度表*或*冷却进度表*，退火进度表设定一个温度的有限序列值，以及每一温度值下有限的转移尝试的次数。Kirkpatrick et al.(1983)给出的退火进度表的感兴值的参数设定如下[4]：

- *温度的初始值*。温度的初始值 T_0 选得足够高使得所有提出的转移实际都能被模拟退火算法所接受。
- *温度的下降*。一般地说，冷却是按指数形式完成的，并且温度值的改变量都很小。特别地，下降函数定义为

$$T_k = \alpha T_{k-1}, \qquad k = 1,2,\cdots \tag{11.34}$$

  其中 α 小于但接近于 1。α 的典型值介于 0.8 和 0.99 之间。对每一温度，有足够的转移的尝试，使得平均每次实验有 10 次转移被接受。
- *温度的最后值*。如果在三次相连的温度下没有得到预期的接收次数，则系统被冻结且退火停止。

后一个标准可以改进，要求*接受率*小于一预定值(Johnson et al.,1989)，而接受率定义为转移接受的次数除以提出转移的次数。

模拟退火用于组合优化

模拟退火特别适用于解组合优化问题。组合优化的目标是针对有很多可能解的有限离散系统，最小化它的代价函数。本质上讲模拟退火利用 Metropolis 算法通过多粒子物理系统和组合优化问题间的类比产生一系列解。

在模拟退火中，我们把式(11.5)的 Gibbs 分布中的能量 E_i 解释成为数值的代价，而温度 T 解释为控制参数。在组合优化问题中对每一构形赋予一数值的代价以描述这个特殊的构形和解的差异。模拟退火程序中下一个需要考虑的问题是如何确认构形和从已有构形以局部方式产生新的构形。这就是 Metropolis 算法发挥作用的地方。因此我们概括统计物理的术语和组合优化术语之间的关系如表 11-1(Beckerman,1997) 560

表 11-1 统计物理与组合优化之间的对应

统计物理	组合优化
样本	问题实例
状态(构形)	构形
能量	代价函数
温度	控制参数
基态能量	最小代价
基态构形	最优构形

11.6 Gibbs 抽样

类似 Metropolis 算法，Gibbs 抽样器[15] 生成一个 Markov 链，它以平衡分布作为 Gibbs 分布。但是 Gibbs 抽样器的转移概率是非平稳的(Geman and Geman,1984)。在最后的分析里，关于 Gibbs 抽样和 Metropolis 算法的选择取决于具体问题的技术细节。

为了继续描述这个抽样格式，考虑一个 K-维的随机向量 $\mathbf{X}$，由分量 X_1，X_2，…，X_K 构成。假定在给定 $\mathbf{X}$ 的其他分量时我们知道 X_k 的条件分布，$k=1$，2，…，K。我们希望问的问题是：对任何 k，怎样获得随机变量 X_k 的边缘密度的数值估计。对随机向量 $\mathbf{X}$ 的每个分量，在已知 $\mathbf{X}$ 的其他分量值的条件下，Gobbs 抽样器对它的条件分布产生一个值。特别地，从任意构形$[x_1(0),x_2(0),\cdots,x_K(0)]$开始，我们在 Gibbs 抽样的第一次迭代时做下列采样：

$x_1(1)$是在已知 $x_2(0)$，$x_3(0)$，…，$x_K(0)$时由 X_1 的分布产生的采样。

$x_2(1)$是在已知 $x_1(1)$，$x_3(0)$，…，$x_K(0)$时由 X_2 的分布产生的采样。

…

$x_k(1)$是在已知 $x_1(1)$，…，$x_{k-1}(1)$，$x_{k+1}(0)$，…，$x_K(0)$时由 X_k 的分布产生的采样。

…

$x_K(1)$是在已知 $x_1(1)$，$x_2(1)$，…，$x_{K-1}(1)$时由 X_K 的分布产生的采样。

在第二次迭代和其他的每次抽样迭代中我们用这种方式进行处理。以下两点需要特别注意

1. 随机向量 $\mathbf{X}$ 的每个分量是以自然序列"访问"的，每次迭代产生总共 K 个新的变量值。

[561] 2. 对于 $k=2$，3，…，K，在对 X_k 采样新值时直接利用分量X_{k-1}的新的值。

由这个讨论我们看到 Gibbs 采样是迭代的自适应格式。利用它进行 n 次迭代后，我们得到 K 个变化量：$X_1(n)$，$X_2(n)$，…，$X_K(n)$。在相当温和的条件下，以下三个定理对 Gibbs 抽样成立(Geman and Geman,1984;Gelfand and Smith,1990)：

1. 收敛定理。对 $k=1$，2，…，K。当 n 趋于无穷大时，随机变量 $X_k(n)$依分布收敛于 X_k 的真实概率分布；也就是说，

$$\lim_{n\to\infty} P(X_k^{(n)} \leqslant x \mid x_k(0)) = F_{X_k}(x), \qquad k=1,2,\cdots,K \tag{11.35}$$

其中 $F_{X_k}(x)$为 X_k 的边缘概率分布函数。

事实上，在 Geman and Geman(1984)中证明了更强的结果。特别地，不要求随机向量 $\mathbf{X}$ 的每个分量以自然顺序被重复访问，任意的访问方式只要不依赖于变量的值且 $\mathbf{X}$ 的每个分量被"无限地经常"访问，则 Gibbs 抽样收敛性仍成立。

2. 收敛速度定理。随机变量 $X_1(n)$，$X_2(n)$，…，$X_K(n)$的联合概率分布以 n 的几何级数速度收敛于 X_1，X_2，…，X_K 的联合分布函数

这个定理假设 $\mathbf{X}$ 的分量以自然顺序访问。但是当使用以任意的但无限地经常访问时，这样收敛速度需要较小的调整。

3. 遍历定理。对任何例如对于随机变量 X_1, X_2, …, X_K 的可测函数 g, 它的期望存在，我们有

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}g(X_1(i),X_2(i),\cdots,X_K(i))\to E[g(X_1,X_2,\cdots,X_K)] \tag{11.36}$$

以概率1(即几乎肯定)实现。

遍历定理告诉我们怎样利用 Gibbs 采样的输出获得所期望的边缘密度的数值估计。

在 Boltzmann 机中使用 Gibbs 采样对有关隐藏神经元的分布进行采样；这种随机机器将在下一节讨论。对于使用二值单元的随机机器(即 Botzmann 机)来说，值得注意的是 Gibbs 采样正好和 Metropolis 算法的一个变体完全一样。在 Metropolis 算法的标准形式里我们以概率 1 下山，相反的在 Metropolis 算法的另一个形式里，我们以 1 或能量差的指数(即上山规则的补充)的概率下山。换句话说，如果一个变化降低了能量 E 或 E 没有变化时，则这个变化被接受；如果变化升高了能量，它是以 $\exp(-\Delta E)$ 的概率被接受，否则被拒绝，而以旧的状态重复(Neal, 1993)。

11.7 Boltzmann 机

Boltzmann 机是由随机神经元组成的随机机器，随机神经元如第 1 章所讨论的那样，以概率方式取两个可能状态之一。这两个状态可以指定为 +1，表示"开"状态，指定为 -1 表示"关"状态，或分别用 1 和 0 表示。我们将采用前面的记号。Boltzmann 机另一个突出的特征就是它的神经元间使用对称的突触连接，这种形式的突触连接也有统计物理方面的考虑。 [562]

Boltzmann 机的随机神经元分成两部分功能组，如图 11-4 所示为可见部分和隐藏部分。可见神经元[6] 提供网络和它运行环境之间的一个界面。在网络的训练阶段，所有可见神经元都被箝制在环境所决定的特定状态。另一方面，隐藏神经元总是自由运行的，它们用来解释环境输入向量包含的固有约束。隐藏神经元通过捕获箝制向量中的高阶统计相关来完成这项任务。这里所叙述的网络代表 Boltzmann 机的一种特殊情况。它可以看成是对某确定概率分布建模的无监督学习程序，该确定概率分布决定于在可见神经元上以合适的概率箝制模式。这样做，网络能起到模式完成(pattern completion)的作用。特别地，当一部分携带信息的向量箝制在可见神经元的子集上，如果网络已经恰当地学会了训练分布，这时网络能够对剩下的可见神经元网络给出它们的恰当的值，起到模式完成的作用(Hinton, 1989)。

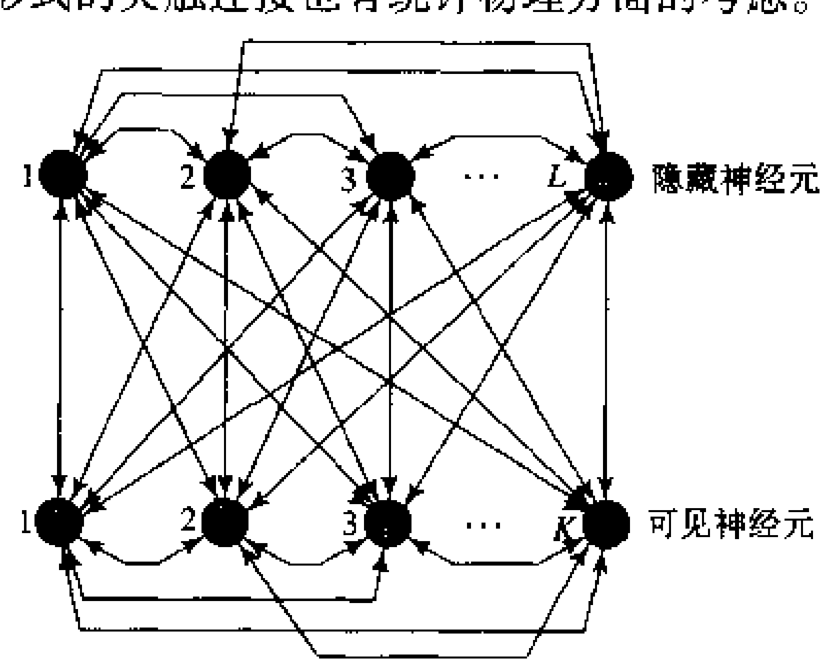


图 11-4 Boltzmann 机体系结构图；K 为可见神经元数目，L 为隐藏神经元数目

Boltzmann 机学习的主要目的是产生一个神经网络，根据 Boltzmann 分布对输入模式进行正确的建模。在这种学习的应用中，作了两个假设：

- 每个环境输入向量(模式)持续足够长的时间，允许网络达到热平衡。
- 环境向量箝制在网络可见单元上的次序是没有任何结构的。

563

一组特定的突触权值当它导出的可见单元状态的概率分布(当网络自由运行时)和可见单元被环境输入向量所箝制时的状态概率分布完全一样，我们说它构造了环境结构的一个完整模型。一般情况下，除非隐藏单元数目是可见单元数目的指数，不可能得到完整模型。但是，如果环境有规则的结构，网络利用隐藏单元捕获这些规则，这时利用较小的能处理的隐藏神经元数目可以对环境取得一个好得匹配。

Boltzmann 机的 Gibbs 抽样和模拟退火

令 $\mathbf{x}$ 表示 Boltzmann 机的状态向量，它的分量 x_i 表示神经元 i 的状态。状态 $\mathbf{x}$ 代表随机向量 $\mathbf{X}$ 的一次实现。从神经元 i 到神经元 j 的突触连接记为 w_{ij}，满足：

$$w_{ji} = w_{ij} \quad 对所有(i,j) \tag{11.37}$$

和

$$w_{ii} = 0 \quad 对所有\ i \tag{11.38}$$

式(11.37)描述对称性而式(11.38)强调无自反馈。偏置可以利用一个输出恒为 +1 的虚节点到神经元 j(对所有 j)的连接权值 w_{j0} 表示。

类似于热动力学，Boltzmann 机的能量可定义为[17]

$$E(\mathbf{x}) = -\frac{1}{2}\sum_{i}\sum_{\substack{j \\ i\neq j}} w_{ji}x_i x_j \tag{11.39}$$

利用(11.5)的 Gibbs 分布，我们可以定义网络(假定处在温度 T 的平衡态)在状态 $\mathbf{x}$ 的概率如下：

$$P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.40}$$

其中 Z 为剖分函数。

为了简化表示，定义单个事件 A 及联合事件 B 和 C 如下：

$$A: X_j = x_j, \quad B:\{X_i = x_i\}_{i=1}^{K}, i\neq j, \quad C:\{X_i = x_i\}_{i=1}^{K}$$

实际上，联合事件 B 排斥 A，而联合事件 C 包括 A 和 B。B 的概率是 C 关于 A 的边缘概率。因此，利用式(11.39)和式(11.40)，我们可写作

564

$$P(C) = P(A,B) = \frac{1}{Z}\exp\left(\frac{1}{2T}\sum_{i}\sum_{\substack{j \\ i\neq j}} w_{ji}x_i x_j\right) \tag{11.41}$$

$$P(B) = \sum_{A} P(A,B) = \frac{1}{Z}\sum_{x_j}\exp\left(\frac{1}{2T}\sum_{i}\sum_{\substack{j \\ i\neq j}} w_{ji}x_i x_j\right) \tag{11.42}$$

在式(11.41)和式(11.42)中的指数可以表示成两项之和，一项和 x_j 有关而另一项与 x_j 无关。包含有 x_j 的项为

$$\frac{x_j}{2T}\sum_{\substack{i \\ i\neq j}} w_{ji}x_i$$

相应地，给定 B，置 $x_j = x = \pm 1$，我们可以给出 A 的条件概率

$$P(A \mid B) = \frac{P(A,B)}{P(B)} = \frac{1}{1+\exp\left(-\frac{x_j}{T}\sum_{\substack{i \\ i\neq j}} w_{ji}x_i\right)}$$

也就是可写成

$$P(X_j = x \mid \{X_i = x_i\}_{i=1, i\neq j}^{K}) = \varphi\left(\frac{x}{T}\sum_{\substack{i \\ i\neq j}} w_{ji}x_i\right) \tag{11.43}$$

其中 $\varphi(\cdot)$ 为它变元的 sigmoid 函数，表示为

$$\varphi(v) = \frac{1}{1+\exp(-v)} \tag{11.44}$$

565

注意 x 虽然在 $+1$ 和 -1 间变化，但当 v 充分大时，整个变量 $v = \frac{x}{T}\sum_{i\neq j} w_{ji}x_i$ 可在 $-\infty$ 和 $+\infty$ 之间变化，如图 11-5 所描绘。同时注意，在推导式(11.43)时，不需剖分函数 Z，这是高度期望的，因为对于非常复杂的网络直接计算 Z 是不现实的。

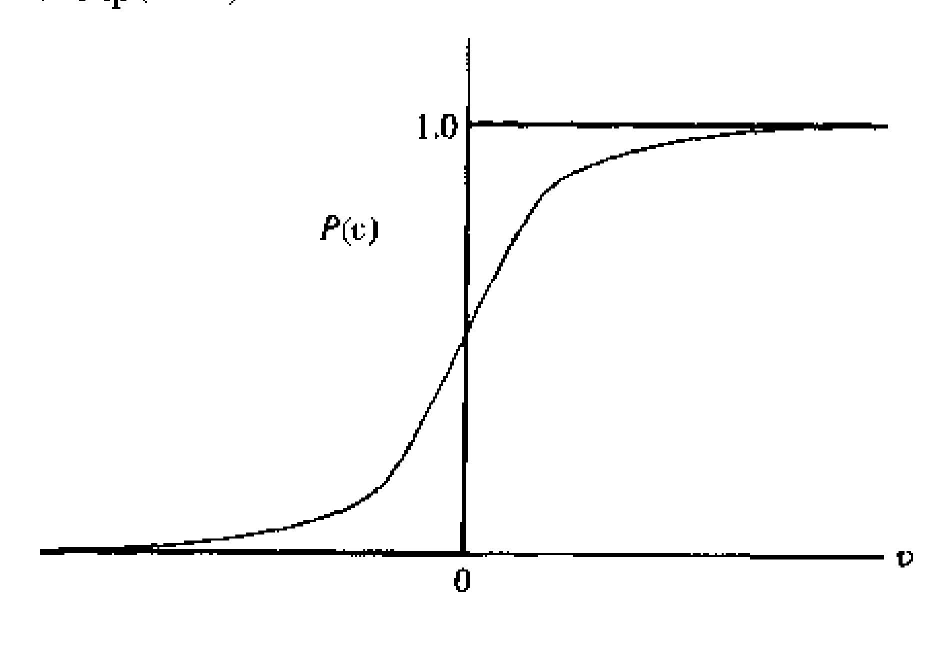


图 11-5　sigmoid - 形函数 $P(v)$

利用 Gibbs 抽样表示联合分布 $P(A, B)$。基本上，如 11.6 节所解释的那样，这个随机模拟开始时给网络赋予任一状态，神经元以它们的自然顺序依次重复访问，每次访问，选择一个神经元，根据其他神经元的值确定该神经元状态新值的选择概率。假定这个随机模拟进行足够长的时间，则网络将达到在温度 T 下的平衡。

遗憾的是到达热平衡的时间可能非常长。为了克服这个困难，如同在 11.5 节所解释的那样，对有限温度序列 T_0，T_1，…，T_{final}，使用模拟退火。特别地，温度被初始化为一个高的值 T，因此可迅速到达热平衡。然后，温度 T 逐渐降低至最后值 T_{final}，这时神经元状态将(有希望)达到它们的边缘分布。

Boltzmann 学习规则

因为 Boltzmann 机是一种随机机器，它自然依赖于用概率论评价其性能。这种标准之一是似然函数[8]。在此基础上，根据最大似然原则，Boltzmann 学习的目标是最大化似然函数或等价的对数似然函数。

令 $\mathscr{T}$ 表示感兴趣的概率分布抽样所组成的训练样本。假设它们都是二值的。训练样本允许重复，但必须和它们发生的概率成比例。令状态向量 $\mathbf{x}$ 的子集 $\mathbf{x}_\alpha$ 表示可见神经元状态。向量 $\mathbf{x}$ 的剩余部分 $\mathbf{x}_\beta$ 表示隐藏神经元的状态。状态向量 $\mathbf{x}$，$\mathbf{x}_\alpha$ 和 $\mathbf{x}_\beta$ 分别表示随机向量 $\mathbf{X}$，$\mathbf{X}_\alpha$ 和 $\mathbf{X}_\beta$ 的实现。Boltzmann 机的运行分成两个阶段：

- 正向阶段。此时网络在箝制环境下(即在训练集 $\mathscr{T}$ 的直接影响下)运行。
- 负向阶段。在第二阶段，网络允许自由运行，因此没有环境输入。

对整个网络给定突触间权值 $\mathbf{w}$，可见神经元状态为 $\mathbf{x}_\alpha$ 的概率是 $P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)$。训练集 $\mathscr{T}$ 中包含许多可能值 $\mathbf{x}_\alpha$，假定它们是统计独立的，总体的概率分布是析因分布 $\prod_{\mathbf{x}_\alpha\in\mathscr{T}} P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)$。为了写出对数似然函数 $L(\mathbf{w})$，对析因分布取对数且将 $\mathbf{w}$ 看作未知的参数向量。我们因此可以写成

$$L(\mathbf{w}) = \log \prod_{\mathbf{x}_\alpha\in\mathscr{T}} P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \sum_{\mathbf{x}_\alpha\in\mathscr{T}} \log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) \tag{11.45}$$

566

为了通过能量函数形成边缘概率 $P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)$ 的表达式，利用以下两点：

- 由式(11.40)，概率 $P(\mathbf{X}=\mathbf{x})$ 等于 $\frac{1}{Z}\exp(-E(\mathbf{x})/T)$。
- 由定义，状态向量 $\mathbf{x}$ 是属于可见神经元的状态 $\mathbf{x}_\alpha$ 和属于隐藏神经元的状态 $\mathbf{x}_\beta$ 的联立组合。因此可见神经元处于状态 $\mathbf{x}_\alpha$ 与任何 $\mathbf{x}_\beta$ 的概率为

$$P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \frac{1}{Z}\sum_{\mathbf{x}_\beta}\exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.46}$$

其中随机向量 $\mathbf{X}_\alpha$ 是 $\mathbf{X}$ 的子集，剖分函数 Z 定义为(参看式(11.6))

$$Z = \sum_{\mathbf{x}}\exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.47}$$

因而将式(11.46)和(11.47)代入式(11.45)，得出对数似然函数所期望的表达式：

$$L(\mathbf{w}) = \sum_{\mathbf{x}_\alpha\in\mathcal{T}}\left(\log\sum_{\mathbf{x}_\beta}\exp\left(-\frac{E(\mathbf{x})}{T}\right) - \log\sum_{\mathbf{x}}\exp\left(-\frac{E(\mathbf{x})}{T}\right)\right) \tag{11.48}$$

对 $\mathbf{w}$ 的依赖包含在能量函数 $E(\mathbf{x})$ 中，如式(11.39)所示。

依据式(11.39)，求 $L(\mathbf{w})$ 对 w_{ji} 的微分，经过一些运算后我们得到下列结果(参看习题11.8)：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \frac{1}{T}\sum_{\mathbf{x}_\alpha\in\mathcal{T}}\left(\sum_{\mathbf{x}_\beta}P(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)x_j x_i - \sum_{\mathbf{x}}P(\mathbf{X} = \mathbf{x})x_j x_i\right) \tag{11.49}$$

为了简单起见，我们引入两个定义：

$$\rho_{ji}^{+} = \langle x_j x_i \rangle^{+} = \sum_{\mathbf{x}_\alpha\in\mathcal{T}}\sum_{\mathbf{x}_\beta}P(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)x_j x_i \tag{11.50}$$

567 和

$$\rho_{ji}^{-} = \langle x_j x_i \rangle^{-} = \sum_{\mathbf{x}_\alpha\in\mathcal{T}}\sum_{\mathbf{x}}P(\mathbf{X} = \mathbf{x})x_j x_i \tag{11.51}$$

从宽松意义上我们可以将第一项平均值 ρ_{ji}^{+} 看成点火率的平均，或神经元 i 和 j 的状态之间的相关性，此时网络在箝制下运行或者说处于正向阶段。类似地，第二项均值 ρ_{ji}^{-} 可看成神经元 i 和 j 的状态间的相关性，此时网络自由运行或者说是处于负向阶段。利用这些定义，我们可以简化式(11.49)如下：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \frac{1}{T}(\rho_{ji}^{+} - \rho_{ji}^{-}) \tag{11.52}$$

Boltzmann 机学习的目的是最大化对数似然函数 $L(\mathbf{w})$，我们可以利用梯度下降法达到这一点，写成

$$\Delta w_{ji} = \epsilon\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \eta(\rho_{ji}^{+} - \rho_{ji}^{-}) \tag{11.53}$$

其中 η 是学习率参数；它通过 ϵ 和运行温度 T 定义为

$$\eta = \frac{\epsilon}{T} \tag{11.54}$$

式(11.53)的梯度下降规则称为 Boltzmann 学习规则。这里所叙述的学习是集中完成的；即突触权值的改变是在整个训练样本集都给出的情况下进行的。

根据这个学习规则，Boltzmann 机的突触权值的调整仅使用两个不同条件下的局部可观测量，这两个不同条件为(1)箝制运行，和(2)自由运行。这个 Boltzmann 学习的重要特征极大地简化了网络结构，特别在处理大型网络时更是如此。另一个重要特征是神经元 i 和 j 之间

的突触权值的调整规则是独立于神经元的可见与否的，不管它们可见或都不可见，这一点可能令人吃惊。Boltzmann 学习的所有这些有益的特征归功于 Hinton and Sejnowski(1983,1986)的关键性见解，它们将 Boltzmann 机的抽象数学模型和神经元网络在以下两点上联系起来：

- 描述一个神经元的随机性的 Gibbs 分布。
- 定义 Gibbs 分布的基于统计物理学的能量函数(11.39)。

从学习观点看，组成 Boltzmann 学习规则的式(11.53)的两项具有相反的意思。我们可以把相应于网络箝制条件下的第一项从本质上看作 Hebb 学习规则，而把相应于网络自由运行下的第二项看作*非学习项或遗忘项*。确实地，Boltzmann *学习规则代表重复遗忘和再学习规则的推广*，这个工作是 Pöppel and Krey(1987)对无隐藏神经元的对称网络所描述的。

既然 Boltzmann 机学习算法要求隐藏神经元知道被刺激和自由活动之间的差异，并且假定有一个(隐藏的)外部网络向隐藏神经元发信号告知 Boltzmann 机正被刺激，我们就有一个注意机制的原始形式(Cowan and Sharp,1988)，这一点倒是很有趣的。 568

负向阶段的需求及其隐含的意义

正向和负向阶段的联合使用稳定 Boltzmann 机突触权值的分布。这种要求可以用另外的方式进行说明。直观上讲，我们可以说在 Boltzmann 学习过程中对正向和负向阶段的要求归因于神经元状态向量的概率表达式中的剖分函数 Z 的出现。这样说暗示着能量空间的最速下降方向和概率空间的最速下降方向不一致。实际上，学习过程的负向阶段需要考虑到这种差异(Neal,1992)。

在 Boltzmann 学习中使用负向阶段有两个主要缺点：

1. *增加计算时间*。在正向阶段，一些神经元由外界环境所箝制，而在负向阶段，所有神经元都自由运行。相应地，Boltzmann 机的随机模拟时间增加了。

2. *对统计误差敏感*。Boltzmann 学习规则涉及两个平均相关性之间的差异，一个相关性计算正向阶段而另一个计算负向阶段。当这两个相关性相似时，抽样噪声的出现使得它们的差异具有更多的噪声。

我们可以利用 sigmoid 信度网络消除 Boltzmann 机的这个缺点，在这类新的随机机器里，对学习过程的控制是利用均值而不是负向阶段。

11.8　sigmoid 信度网络

sigmoid 信度网络或 logistic 信度网络由 Neal 在 1992 年所发展的，它主要是为了寻找一种随机机器，它既享有 Boltzmann 机能学习任何二值概率分布的能力，但不需要 Boltzmann 机学习过程的负向阶段。这个目标的达到，是用*有向连接构成的无圈图*代替 Boltzmann 机的对称连接。特别地，一个 sigmoid 信度网络由二值随机神经元的多层结构组成，如图 11-6 所示。机器具有无圈的性质使得概率计算简单。尤其是，类似于 Boltzmann 机，网络利用式(11.43)的 sigmoid 函数计算一个神经元受到它自己的诱导局部域刺激时的条件概率。

sigmoid 信度网络的基本性质

令向量 $\mathbf{X}$ 由二值随机变量 X_1，X_2，…，X_N 组成，它定义由 N 个随机神经元构成的一个 sigmoid 信度网络。在 $\mathbf{X}$ 中的元素 X_j 的*双亲*记为

569

$$\mathrm{pa}(X_j) \subseteq \{X_1, X_2, \cdots, X_{j-1}\} \tag{11.55}$$

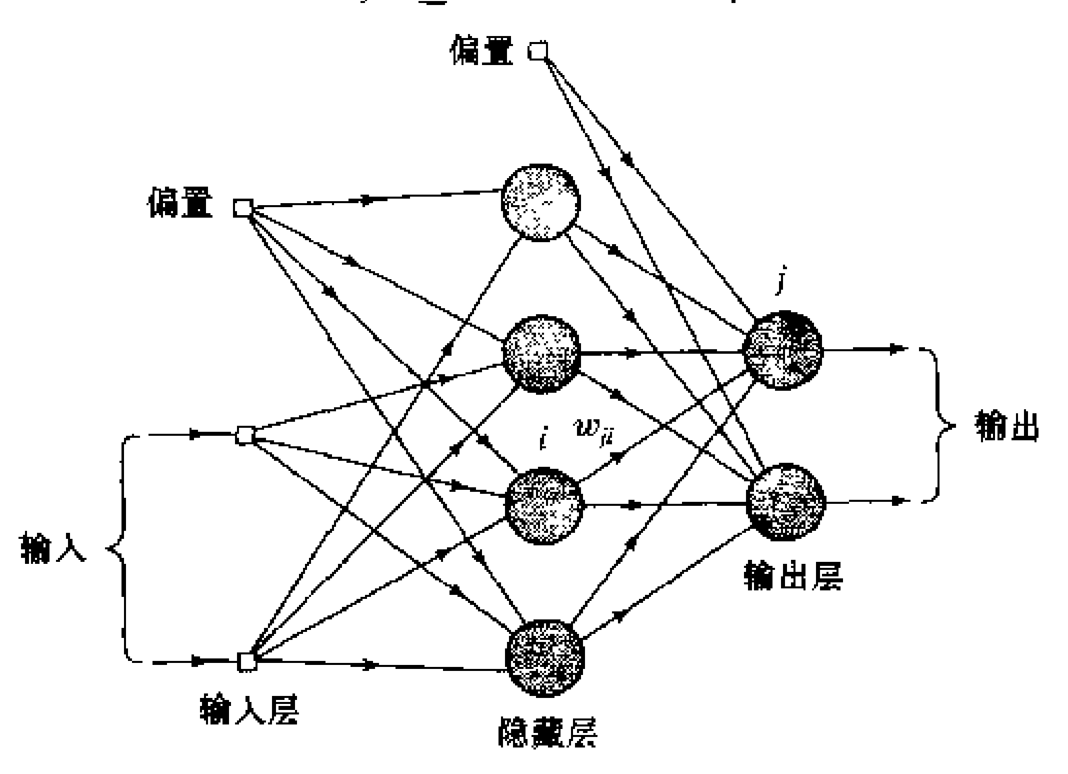


图 11-6 sigmoid 信度网络结构图

也就是说，$\mathrm{pa}(X_j)$是随机向量 **X** 满足下式的最小子集：

$$P(X_j = x_j \mid X_1 = x_1, \cdots, X_{j-1} = x_{j-1}) = P(X_j = x_j \mid \mathrm{pa}(X_j)) \tag{11.56}$$

sigmoid 信度网络的一个重要优点就是它能清楚揭示输入数据的固有概率模型的条件依赖。特别地，第 j 个神经元被激发的概率由 sigmoid 函数

$$P(X_j = x_j \mid \mathrm{pa}(X_j)) = \varphi\left(\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right) \tag{11.57}$$

定义(参看式(11.43))，其中 w_{ji}是从神经元 i 到神经元 j 的突触权值，如图 11-6 所示。即是条件概率 $P(X_j = x_j \mid \mathrm{pa}(X_j))$仅依赖于 $\mathrm{pa}(X_j)$的输入加权和。因此，式(11.57) 提供信度在网络中传播的基础。

在 sigmoid 信度网络中计算概率，以下两点值得注意：

1. $w_{ji} = 0$，对所有不属于 $\mathrm{pa}(X_j)$的 X_i

2. $w_{ji} = 0$，对所有 $i \geqslant j$

第一点由双亲的定义可得。第二点由 sigmoid 信度网络是有向无圈图这个事实可得。

正如名字所暗示，sigmoid 信度网络属于在文献(Pearl, 1988)中被广泛研究的一类信度网络[9]。它的随机运行比 Boltzmann 机稍微复杂一些。然而基于局部可用信息，它们确实可以利用概率空间的梯度上升学习。

570

sigmoid 信度网络的学习

令$\mathcal{T}$表示以感兴趣的概率分布抽取的训练样本集。假定每一个样本都是二值的，表示一定的属性。训练样本是允许重复的，重复的次数与已知的特定属性组合通常发生的概率成正比。为了对从其中抽取$\mathcal{T}$的分布进行建模，我们作如下处理：

1. 用一个状态向量 **x** 的大小决定网络的大小。

2. 选择状态向量的一个子集，记为 $\mathbf{x}_\alpha$，代表训练时的属性，即 $\mathbf{x}_\alpha$ 代表可见神经元(即证据节点)的状态向量。

3. 用状态向量 $\mathbf{x}$ 的剩余部分，记为 $\mathbf{x}_\beta$，定义为隐藏神经元(即那些我们没有瞬时值的计算节点)的状态向量。

对于给定的状态向量 $\mathbf{x}$，一个 sigmoid 信度网络的设计高度依赖于可见神经元和隐藏神经元的排列方式。因此可见神经元和不可见神经元的不同排列会导致不同的构形。

正如 Boltzmann 机一样，我们导出 sigmoid 信度网络所期望的学习规则时仍然最大化对数似然函数，而对数似然函数可由训练集$\mathscr{T}$计算可得。由式(11.45)定义的对数似然函数 $L(\mathbf{w})$，为表达方便重写如下：

$$L(\mathbf{w}) = \sum_{\mathbf{x}_\alpha \in \mathscr{T}} \log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)$$

其中 $\mathbf{w}$ 为网络的突触权值向量，作为未知的处理。属于可见神经元的状态向量 $\mathbf{x}_\alpha$ 是随机向量 $\mathbf{X}_\alpha$地一次实现。令 w_{ji} 表示 $\mathbf{w}$ 的第 ji 个元素(即从神经元 i 到神经元 j 的突触权值)。对 $L(\mathbf{w})$求 w_{ji} 的微分，我们有

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \sum_{\mathbf{x}_\alpha \in \mathscr{T}} \frac{1}{P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)} \frac{\partial P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)}{\partial w_{ji}}$$

下一步我们注意下列两个概率关系：第一个关系

$$P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \sum_{\mathbf{x}_\beta} P(\mathbf{X} = (\mathbf{x}_\alpha, \mathbf{x}_\beta)) = \sum_{\mathbf{x}_\beta} P(\mathbf{X} = \mathbf{x}) \tag{11.58}$$

其中随机向量 $\mathbf{X}$ 属于整个网络，而状态向量 $\mathbf{x} = (\mathbf{x}_\alpha, \mathbf{x}_\beta)$是它的一次实现。第二个关系

$$P(\mathbf{X} = \mathbf{x}) = P(\mathbf{X} = \mathbf{x} \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha) P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) \tag{11.59}$$

571

这个关系定义联合事件 $\mathbf{X} = \mathbf{x} = (\mathbf{x}_\alpha, \mathbf{x}_\beta)$的概率。

根据这两个关系，我们可以重新定义偏导数$\partial L(\mathbf{w})/\partial w_{ji}$的等价形式：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \sum_{\mathbf{x}_\alpha \in \mathscr{T}} \sum_{\mathbf{x}_\beta} \frac{P(\mathbf{X} = \mathbf{x} \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)}{\mathrm{P}(\mathbf{X} = \mathbf{x})} \frac{\partial P(\mathbf{X} = \mathbf{x})}{\partial w_{ji}} \tag{11.60}$$

根据式(11.43)我们可写成

$$P(\mathbf{X} = \mathbf{x}) = \prod_j \varphi\left(\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right) \tag{11.61}$$

其中 $\varphi(\cdot)$为 sigmoid 函数。因此可写成

$$\begin{aligned}\frac{1}{P(\mathbf{X} = \mathbf{x})} \frac{\partial P(\mathbf{X} = \mathbf{x})}{\partial w_{ji}} &= \frac{\partial}{\partial w_{ji}} \log P(\mathbf{X} = \mathbf{x}) = \frac{\partial}{\partial w_{ji}} \sum_j \log \varphi\left(\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right) \\ &= \frac{1}{T} \sum_j \frac{1}{\varphi\left(\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right)} \varphi'\left(\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right) x_j x_i\end{aligned}$$

其中 $\varphi'(\cdot)$表示 sigmoid 函数 $\varphi(\cdot)$关于它的变量的一阶导数。但是，从(11.44)给出的 $\varphi(\cdot)$的定义，容易发现

$$\varphi'(v) = \varphi(v)\varphi(-v) \tag{11.62}$$

其中 $\varphi(-v)$是由 $-v$ 替代 $\varphi(v)$中的 v 而得到的。因此，我们可写成

$$\frac{1}{P(\mathbf{X} = \mathbf{x})} \frac{\partial P(\mathbf{X} = \mathbf{x})}{\partial w_{ji}} = \frac{1}{T} \sum_j \varphi\left(-\frac{x_j}{T} \sum_{i<j} w_{ji} x_i\right) x_j x_i \tag{11.63}$$

相应地，将式(11.63)代入式(11.60)，我们得到

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \frac{1}{T}\sum_{\mathbf{x}_\alpha \in \mathcal{T}}\sum_{\mathbf{x}_\beta} P(\mathbf{X} = \mathbf{x} \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\varphi\left(-\frac{x_j}{T}\sum_{i<j} w_{ji}x_i\right)x_j x_i \tag{11.64}$$

为简单起见，我们定义整体均值

$$\rho_{ji} = \left\langle \varphi\left(-x_j \sum_{i<j} w_{ji}x_i\right)x_j x_i \right\rangle$$

572

$$= \sum_{\mathbf{x}_\alpha \in \mathcal{T}}\sum_{\mathbf{x}_\beta} P(\mathbf{X} = \mathbf{x} \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\varphi\left(-\frac{x_j}{T}\sum_{i<j} w_{ji}x_i\right)x_j x_i \tag{11.65}$$

它代表神经元 i 和 j 状态的平均相关性乘以加权因子 $\varphi\left(-\frac{x_j}{T}\sum_{i<j} w_{ji}x_i\right)$。这个平均是对所有 $\mathbf{x}_\alpha$ 的可能值(由训练集$\mathcal{T}$中抽取)及 $\mathbf{x}_\beta$ 的所有可能值求得的，这里 $\mathbf{x}_\alpha$ 表示可见神经元而 $\mathbf{x}_\beta$ 表示隐藏神经元。

在概率空间中的梯度上升可以由定义突触权值 w_{ji} 的增量改变

$$\Delta w_{ji} = \epsilon \frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \eta \rho_{ji} \tag{11.66}$$

来完成，其中 $\eta = \epsilon / T$ 为学习速度参数，ρ_{ji} 由式(11.65)定义。式(11.66)为 sigmoid 信度网络的学习规则。

sigmoid 信度网络学习过程的小结由表 11-2 给出，其中学习是以集中方式完成的，即网络突触权值的改变是基于整个训练集作出的。由表 11-2 给出的小结不包括对模拟退火的使用，这也是我们置温度 T 等于 1 的原因。但是，正如在 Boltzmann 机一样，如果期望 sigmoid 信度网络学习过程更快到达热平衡，则在学习程序中可以结合模拟退火。

表 11-2　sigmoid 信度网络学习过程小结

初始化。初始化网络，设置网络权值 w_{ji} 为$[-a, a]$区间内均匀分布的随机数；a 的一个典型值为 0.5。

1. 给定训练例子集$\mathcal{T}$，箝制网络的可见神经元到 $\mathbf{x}_\alpha$，其中 $\mathbf{x}_\alpha \in \mathcal{T}$。
2. 对每一个 $\mathbf{x}_\alpha$，在某个运行温度 T 下执行网络单独的 Gibbs 采样模拟，并观察整个网络的状态向量 $\mathbf{x}$ 的结果。假设执行的模拟时间足够长，对于训练集$\mathcal{T}$中的不同例子，$\mathbf{x}$ 的取值应该来自给定训练集对应的随机向量 $\mathbf{X}$ 的条件分布。
3. 计算总体平均值

$$\rho_{ji} = \sum_{\mathbf{x}_\alpha \in \mathcal{T}}\sum_{\mathbf{x}_\beta} P(\mathbf{X} = \mathbf{x} \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)x_j x_i \varphi\left(-x_j \sum_{i<j} w_{ji}x_i\right)$$

其中随机向量 $\mathbf{X}_\alpha$ 是 $\mathbf{X}$ 的子集，且 $\mathbf{x} = (\mathbf{x}_\alpha, \mathbf{x}_\beta)$，$\mathbf{x}_\alpha$ 表示可见神经元，$\mathbf{x}_\beta$ 表示隐藏神经元；x_j 是状态向量 $\mathbf{x}$ 的第 j 个元素(即神经元 j 的状态)，w_{ji} 为神经元 i 到神经元 j 的突触权值。sigmoid 函数 $\varphi(\cdot)$ 定义为

573

$$\varphi(v) = \frac{1}{1 + \exp(-v)}$$

4. 网络的每个突触权值 w_{ji} 的增加量为 $\Delta w_{ji} = \eta \rho_{ji}$，其中 η 是学习率参数。根据最大似然原则，这种调整将沿梯度移动网络的突触权值到似然函数 $L(\mathbf{w})$ 的一个局部最大值。

与 Boltzmann 机不同，在 sigmoid 信度网络学习中仅需一个阶段。这样简化是因为状态向量的概率分布的归一化由 sigmoid 函数 $\varphi(\cdot)$ 对每个神经元局部完成，而不经过计算涉及所有可能的状态构形剖分函数 Z 的困难。由训练集$\mathcal{T}$中抽取给定的 $\mathbf{x}_\alpha$ 的值，一旦随机向量 $\mathbf{X}$ 的条件分布已经由 Gibbs 抽样正确地建模，在 Boltzmann 学习过程的负向阶段所起的作用就被加

权因子 $\varphi\left(-\frac{x_i}{T}\sum_{i<j} w_{ji}x_i\right)$ 所替代，它涉及计算神经元 i 和 j 的状态间的总体平均相关性 ρ_{ji}。当达到对数似然函数 $L(\mathbf{w})$ 的局部最小值时，这时如果网络学习的是确定性的映射，则加权因子将变为零；否则它的平均作用效果将不为零。

在 Neal(1992)的实验结果表明，(1)sigmoid 信度网络能够对非平凡的分布模型进行模拟学习，(2)这些网络能够比 Boltzmann 机有更快的学习率，(3)sigmoid 信度网络对 Boltzmann 机的这个优点是因为消除学习过程中的负向阶段。

11.9 Helmholtz 机

sigmoid 信度网络提供一个强有力的多层框架，用无监督的方式表示和学习我们感兴趣的感知输入中的高阶统计关系。由 Dayan et al.(1995)和 Hinton et al.(1995)首先描绘的 Helmholtz[10] 机提供另一个精巧的多层框架，可以不用 Gibbs 抽样而达到同样的目的。

Helmholtz 机使用两组完全不同的突触连接集，如图 11-7 表示的两层的二值随机神经元网络。在图 11-7 中的实线表示前向连接，它们构成识别模型。这个模型的目的是推断引起输入向量的固有概率分布。在图 11-7 中的虚线表示反向连接，它们构成产生模型。第二个模型的目的是从网络隐藏层所捕获的固有表示中重构对原始输入向量的逼近，从而使之能以自监督的方式运行。识别模型和产生模型以严格的前馈方式运行，没有反馈；它们只在学习过程中相互作用。

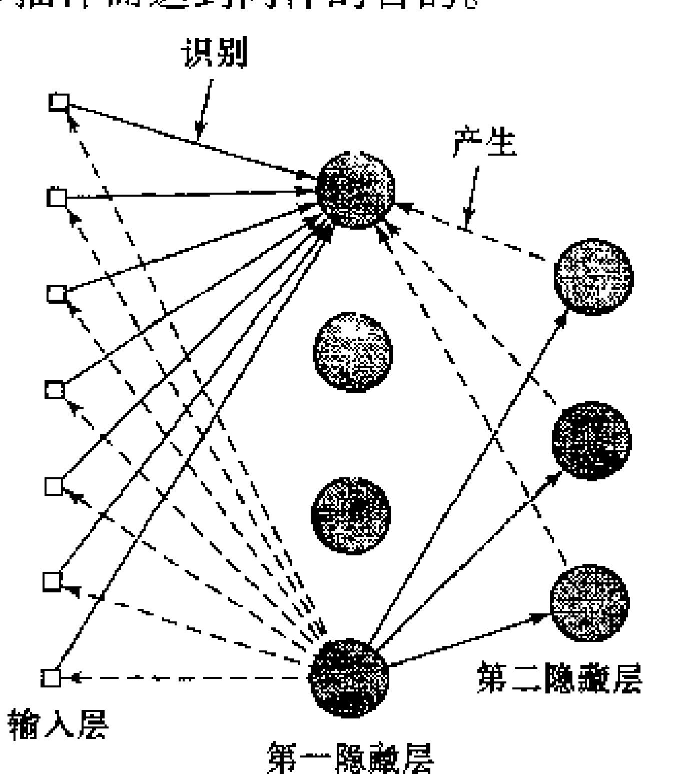


图 11-7　由识别(实线)连接和产生(虚线)连接的互连神经元构成的 Helmholtz 机结构图

574

Hinton et al.(1995)描述一个称为“唤醒－休眠”算法计算 Helmholtz 机的识别权值和产生权值。正如名字所提示的，算法分两个阶段：一个“唤醒”阶段和一个“休眠”阶段。在“唤醒”阶段，网络由识别权值用前向方式驱动。因此在第一个隐层产生一个输入向量的表示。接着第二个隐层产生对第一个表示的表示，对其他网络隐藏层依此类推。网络不同隐藏层产生的表示集提供网络对输入向量的总体表示。虽然此时神经元是由识别模型的权值驱动，但在“唤醒”阶段只有产生模型的权值利用局部可用信息进行学习。实际上，学习过程的这个阶段使得总体表示的每一层在重建前一层形成的激活中都得到提高。

在算法“休眠”阶段，识别模型的权值被禁止。网络由产生权值反向驱动，从最外面的隐藏层开始，逐层反向运行直至输入层。由于神经元是随机的这个事实，重复这个过程一般会在输入层产生许多不同的“幻想”向量。这些幻想提供网络产生模型关于世界的一个无偏抽样。产生一个“幻想”之后，利用简单的 delta 规则(在第 3 章描述)调整识别权值，使得引起“幻想”的隐藏活动的恢复概率的对数最大化。如同“唤醒”阶段一样，“休眠”阶段仅利用局部可用信息。

产生权值(即反向连接)的学习规则依然使用简单的 delta 规则。但是，这个规则沿着一种惩罚对数似然函数的梯度而不是对数似然函数的梯度。惩罚项是真实的后验分布和识别模

型所生成的实际分布之间的 Kullback-Leibler 散度(Hinton et al., 1995); Kullback-Leibler 散度或相对熵在前一章里曾经讨论过。实际上，惩罚对数似然函数是输入数据对数似然函数的一个下界，通过学习过程提高这个下界。特别地，学习过程试图调整产生权值使得近似真实的后验分布尽可能地靠近识别模型实际计算的分布。很遗憾，学习识别模型的权值并不是精确地对应于惩罚似然函数。唤醒 - 休眠学习过程不能保证在所有实际场合都成功；有时它会失败。

575

11.10 平均场理论

前三节所考虑的学习机器有一个共同的特征：它们都使用随机神经元，因此可能导致学习过程很缓慢。在本章的第三部分和最后部分，我们研究利用平均场理论为数学基础导出这些随机机器的*确定性逼近*以加速学习。由于这里讨论的随机机器有不同的结构，相应地使用平均场理论的方式也不相同。特别地，我们可以验证在文献中被研究过的两种特殊方法：

1. 相关性用它们的平均场逼近替代；
2. 通过变分原理用一个易解模型替代一个难解模型。

方法 2 是高度原则化的，因此非常有吸引力。它适用于 sigmoid 信任网(Saul et al.1996)和 Helmholtz 机(Dayan et al.,1995)。但是应用方法 2 到 Boltzmann 机时非常复杂，因为需要剖分函数 Z 的一个上界。由于这个原因，Peterson and Anderson(1987)应用第一个方法加速 Boltzmann 学习规则。在这一节我们为第一种方法提供理论基础，第二种方法在本章后面考虑。

平均场逼近的思想在统计物理学中是熟知的(Glauber,1963)。虽然不能否认在随机机器的背景下期望在所有时刻知道网络中所有神经元的状态，但是，我们必须承认，在神经元数目比较大的网络中，神经状态包含比我们实际所需要的多得多的信息。事实上，我们仅需要知道神经元状态的均值或神经状态对的乘积的均值。

在一个随机神经元里，点火机制由随机规则描述。在这种情况下，对我们而言一个合理的要求就是查询神经元 j 的状态 x_j 的均值。精确地说，这个均值为一种"热"平均，因为突触噪声常常根据热波动建模。对任何事件，令 $<x_j>$ 表示 x_j 的均值。神经元 j 的状态由概率规则

$$x_j = \begin{cases} +1 & \text{以概率 } P(v_j) \\ -1 & \text{以概率 } 1 - P(v_j) \end{cases} \tag{11.67}$$

描述，其中

$$P(v_j) = \frac{1}{1 + \exp(-v_j/T)} \tag{11.68}$$

式中 T 为运行温度。因此我们可以利用给定的诱导局部域 v_j 的特定值表示均值 $<x_j>$ 如下：

$$\begin{aligned} <x_j> &= (+1)P(v_j) + (-1)[1 - P(v_j)] \\ &= 2P(v_j) - 1 \\ &= \tanh(v_j/2T) \end{aligned} \tag{11.69}$$

576

其中 $\tanh(v_j/2T)$是$(v_j/2T)$的双曲正切函数。图 11-8 给出均值 $<x_j>$ 对诱导局部域 v_j 两种图。连续曲线对应于大于零的某个温度 T，粗实线对应于 $T=0$ 的极限情况。在后一种情况，式(11.69)取极限形式

$$<x_j> \to \mathrm{sgn}(v_j) \quad \text{当 } T \to 0 \tag{11.70}$$

这对应于 McCulloch-Pitts 神经元的激活函数。

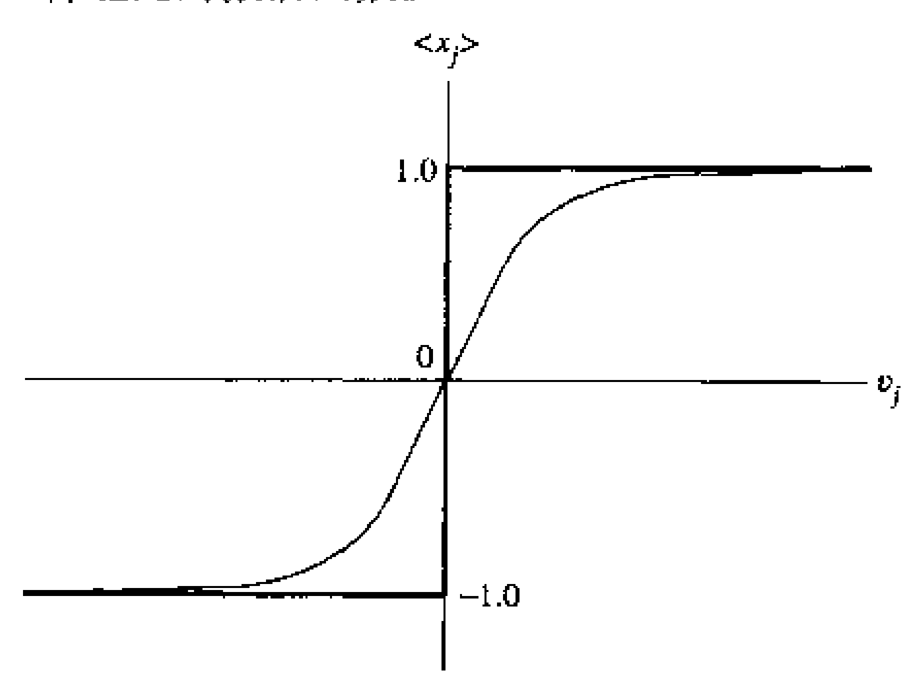


图 11-8　热平均 $<x_j>$ 关于诱导局部域 v_j 的图；粗实曲线对应于 McCulloch-Pitts 神经元的常规操作

到目前为止，讨论集中在单个随机神经元这种简单情形。对于更常见的情形，由大量神经元组成的随机机器，这是一个困难得多的任务。出现困难归因于以下两个因素的组合：

- 神经元 j 的概率 $P(v_j)$ 是诱导局部域 v_j 的非线性函数。
- 诱导局部域 v_j 是一个随机变量，它受到和神经元 j 的输入相连接的其他神经元的随机活动的影响。

大体上可以有把握地说，我们还没有可以利用的数学方法使之精确评价随机机器的行为。但我们可以利用已知的通称为*平均场逼近*的近似方法，它常常产生良好的结果。平均场逼近的基本思想是对网络中每个神经元 j 用诱导局部域 v_j 的平均替代神经波动 v_j，可表示为

$$v_j \overset{\text{approx}}{=} < v_j > = \left\langle \sum_i w_{ji} x_i \right\rangle = \sum_i w_{ji} < x_i > \tag{11.71}$$

577

因此，我们可以计算由 N 个神经元构成的随机机器的第 j 个神经元的平均状态 $< v_j >$，正如在式(11.69)对单个随机神经元所做的那样，可写为

$$< x_j > = \tanh\left(\frac{1}{2T} v_j\right) \overset{\text{approx}}{=} \tanh\left(\frac{1}{2T} < v_j >\right) = \tanh\left(\frac{1}{2T} \sum_i w_{ji} < x_j >\right) \tag{11.72}$$

依据式(11.72)，我们可以正式陈述平均场逼近如下：

一个随机变量某个函数的平均用随机变量平均的函数逼近。

对 $j=1, 2, \cdots, N$，式(11.72)表示具有 N 个未知量 $< x_j >$ 的非线性方程组。这个非线性方程组的解是一个易处理的命题，因为未知量是确定的而不像在原来网络中它们是随机变量。

11.11　确定性的 Boltzmann 机

Boltzmann 机学习与神经元数目成指数关系，因为 Boltzmann 学习规则要求计算网络中每一对神经元之间的相关性。因而 Boltzmann 学习需要指数的时间。Peterson and Anderson (1987) 提出了加速 Boltzmann 学习过程的方法。该方法涉及用一种平均场逼近替代 Boltzmann 学习规则式(11.53)中的相关性，可表示为

$$< x_j x_i > \overset{\text{approx}}{=} < x_j > < x_i >, (i,j) = 1,2,\cdots,K \tag{11.73}$$

其中平均量 $< x_j >$ 利用平均场方程(11.72)计算。

利用刚才描述的方式逼近相关性的计算，这种形式的 Boltzmann 学习称为*确定性的 Boltzmann 学习规则*。特别地，标准的 Boltzmann 学习规则式(11.53)被逼近如下：

$$\Delta w_{ji} = \eta(U_j^+ U_i^+ - U_j^- U_i^-) \tag{11.74}$$

其中 U_j^+ 和 U_j^- 分别表示可见神经元 j(在单个模式上)处于箝制和自由运行情况下的平均输出，η 是学习率参数。虽然 Boltzmann 机使用二值的随机神经元，但它的确定性网络却使用类似的确定性神经元。

[578] 确定性的 Boltzmann 机比标准的 Boltzmann 机在学习速度上提高一至两个数量级(Peterson and Anderson, 1987)。但是，在它的实际应用中仍有两点需注意：

1. 确定性的 Boltzmann 学习规则只在监督情况下有效，即当有些可见神经元作为输出神经元时。无监督学习完全不能在平均场领域应用，因为平均状态是自由运行概率分布的一个显著改进的表示。

2. 在监督学习的情况下，使用确定性的 Boltzmann 学习限制在仅含有一个隐藏层的神经网络(Galland, 1993)。从理论上讲，没有任何理由不可以用到多个隐藏层，但在实际上使用多个隐藏层导致和第 1 点中提到的无监督学习一样的问题。

式(11.74)的确定性 Boltzmann 学习规则有一个简单和局部的形式，这使得它易于用超大规模集成电路(VLSI)硬件实现(Alspector et al., 1991; Schneider and Card, 1993)。但是，在 Schneider and Card(1998)中报告电容权值的连续学习时，确定性的 Boltzmann 机不能忍受在学习电路中权值存储电容器改变的延迟和偏差。这是因为这些内部问题导致突触权值偏移，引起振荡，这显然是不能接受的。

11.12 确定性的 sigmoid 信度网络

在 11.10 节描述的平均场逼近的本质在于用随机变量均值的函数逼近随机变量函数的均值。对 Boltzmann 机的逼近，由前一节讨论可知，平均场理论的这个观点只有在限制情况下有用。这一节我们描绘平均场理论的另一个观点，它适合于 sigmoid 信度网络的逼近。基本上，在这里发现对一个难解模型经过变分原理可由一个易解模型进行逼近(Saul et al., 1996; Jordan et al., 1998)。一般说来，易解模型的特点就是降低难解模型的自由度。针对特定问题设计出适宜的所谓*变分参数*，扩展难解模型使之包括这些附加参数，这样就可以完成自由度的降低。这些术语来自植根于变分法技术的使用(Parisi, 1988)。

对数似然函数的下界

我们讨论的出发点是式(11.58)中的概率关系，这里以对数形式重写如下：

$$\log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \log \sum_{\mathbf{x}_\beta} P(\mathbf{X} = \mathbf{x}) \tag{11.75}$$

如同在 11.8 节，我们剖分随机向量 $\mathbf{X}$ 成 $\mathbf{X}_\alpha$ 和 $\mathbf{X}_\beta$，令 $\mathbf{X}_\alpha$ 对应于可见神经元，而 $\mathbf{X}_\beta$ 对应于隐藏神经元。随机向量 $\mathbf{X}$，$\mathbf{X}_\alpha$ 和 $\mathbf{X}_\beta$ 的实现分别记为 $\mathbf{x}$，$\mathbf{x}_\alpha$，$\mathbf{x}_\beta$。现在，式(11.75)中要求概率和的对数是很难处理的。注意对任何条件分布 $Q(\mathbf{X}_\beta = \mathbf{x}_\beta | \mathbf{X}_\alpha = \mathbf{x}_\alpha)$，我们可以将式(11.75)

重写成不同但等价的形式，这样我们就可以克服这个困难：

$$\log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \log \sum_{\mathbf{x}_\beta} Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\left[\frac{P(\mathbf{X} = \mathbf{x})}{Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)}\right] \quad (11.76)$$

579

这个等式写成这种形式是为了应用前一章讨论的 Jensen 不等式。关于这个应用，我们获得下界：

$$\log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) \geqslant \sum_{\mathbf{x}_\beta} Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\log\left[\frac{P(\mathbf{X} = \mathbf{x})}{Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)}\right] \quad (11.77)$$

考虑到平均场理论，今后我们将把逼近分布 $Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)$称为平均场分布。

我们感兴趣的是对数似然函数的公式。在 sigmoid 信度网络时，对数似然函数 $L(\mathbf{w})$的定义是对所有 $\mathbf{x}_\alpha$(由训练集$\mathscr{T}$决定)求和，因而网络使用集中式算法。我们将使用不同策略求 sigmoid 信度网络的平均场逼近。特别是，将采用串行运算方式，对数似然函数的计算是在一个一个例子的基础上进行的，表示为

$$\mathscr{L}(\mathbf{w}) = \log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) \quad (11.78)$$

其中 $\mathbf{w}$ 为网络权值向量。对独立同分布的(iid)数据，实际的对数似然函数$\mathscr{L}(\mathbf{w})$是对每个数据点的$\mathscr{L}(\mathbf{w})$项的和。这样情况下，$L(\mathbf{w})$的定义基本上和$\mathscr{L}(\mathbf{w})$等价。一般利用$\mathscr{L}(\mathbf{w})$可以提供 $L(\mathbf{w})$的一个逼近。

串行或在线学习方式已经变成了神经网络设计的标准方式，这主要由于它的实现简单。因而依据式(11.78)，可以写成

$$\mathscr{L}(\mathbf{w}) \geqslant \sum_{\mathbf{x}_\beta} Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\log\left[\frac{P(\mathbf{X} = \mathbf{x})}{Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)}\right]$$

或等价地，

$$\mathscr{L}(\mathbf{w}) \geqslant -\sum_{\mathbf{x}_\beta} Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\log Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)$$
$$+ \sum_{\mathbf{x}_\beta} Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)\log P(\mathbf{X} = \mathbf{x}) \quad (11.79)$$

580

式(11.79)右边第一项为平均场分布 $Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)$的熵；不要把它和条件熵混淆。第二项是就隐藏神经元的所有可能状态对 $\log P(\mathbf{X} = \mathbf{x})$的求平均。在单位温度，由 11.2 节中对 Gibbs 分布的讨论，注意 sigmoid 信度网络的能量是 $-\log P(\mathbf{X} = \mathbf{x})$。从式(11.61)我们有(对 $T = 1$)

$$P(\mathbf{X} = \mathbf{x}) = \prod_j \varphi\left(x_j \sum_{i<j} w_{ji}x_i\right)$$

随之有

$$E = -\log P(\mathbf{X} = \mathbf{x}) = -\sum_j \log\varphi\left(x_j \sum_{i<j} w_{ji}x_i\right) \quad (11.80)$$

使用 sigmoid 函数的定义

$$\varphi(v) = \frac{1}{1 + \exp(-v)} = \frac{\exp(v)}{1 + \exp(v)}$$

因而可以把 sigmoid 信度网络的能量函数表示为

$$E = -\sum_i \sum_{\substack{j \\ i<j}} w_{ji}x_ix_j + \sum_j \log\left(1 + x_j \sum_{i<j} w_{ji}x_i\right) \quad (11.81)$$

除去一个乘数因子 1/2，式(11.81)的右边第一项可以看成一个 Markov 系统(即 Boltzmann 机)的能量函数，但是第二项对 sigmoid 信度网络是惟一的。

式(11.79)的下界对任何平均场分布 $Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha)$都是对的。但是，为了很好利用

它，必须选择分布使得可以估计这个界。这仅需要选择析因分布(Saul et al.,1996)

$$Q(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha) = \prod_{j\in\mathcal{H}} \mu_j^{x_j}(1-\mu_j)^{1-x_j} \tag{11.82}$$

其中$\mathcal{H}$表示所有隐藏神经元的集合，且它们的状态为独立的具有可调均值μ_j的Bernoulli变量(一个Bernoulli(θ)定义为取值1的概率为θ的二值随机变量)。因此，将(11.82)代入到式(11.79)我们得到(经过化简)：

$$\mathcal{L}(\mathbf{w}) \geqslant -\sum_{j\in\mathcal{H}}[\mu_j\log\mu_j + (1-\mu_j)\log(1-\mu_j)] + \sum_{i}\sum_{\substack{j\in\mathcal{H}\\ i<j}} w_{ji}\mu_i\mu_j - \sum_{j\in\mathcal{H}} \left\langle \log\left[1+\exp\left(\sum_{i<j} w_{ji}x_i\right)\right]\right\rangle \tag{11.83}$$

其中用$\langle\cdot\rangle$表示关于平均场分布的总体平均，$j\in\mathcal{H}$表示j是一个隐藏神经元。式(11.83)右边第一项是平均场熵，第二项为平均场能量。这两项都是关于式(11.82)的析因分布的。

遗憾的是，我们仍然有一个难解问题：精确计算$\langle\log[1+\exp(z_j)]\rangle$形式的均值是不可能的。这项出现在(11.83)中，包含

581

$$z_j = \sum_{i<j} w_{ji}x_i \tag{11.84}$$

为了克服这个困难，我们重新利用Jensen不等式得到一个界。首先，对任何随机变量z_j和任何实数ξ_j，把$\langle\log[1+\exp(z_j)]\rangle$表示成等价的另一种形式

$$\langle\log(1+e^{z_j})\rangle = \langle\log[e^{\xi_j z_j}e^{-\xi_j z_j}(1+e^{z_j})]\rangle = \xi_j\langle z_j\rangle + \langle\log[e^{-\xi_j z_j}+e^{(1-\xi_j)z_j}]\rangle \tag{11.85}$$

其中$\langle z_j\rangle$为z_j的总体平均。其次，和以前使用的Jensen不等式相比，我们反方向使用它，这样可以得到式(11.85)右边关于平均值的一个上界

$$\langle\log(1+e^{z_j})\rangle \leqslant \xi_j\langle z_j\rangle + \log\langle e^{-\xi_j z_j}+e^{(1-\xi_j)z_j}\rangle \tag{11.86}$$

在式(11.86)中置$\xi_j=0$，我们获得标准界

$$\langle\log(1+e^{z_j})\rangle \leqslant \log\langle 1+e^{z_j}\rangle$$

在式(11.86)中允许使用非零值ξ_j，可得均值$\langle\log(1+e^{z_j})\rangle$的一个可能比标准界更紧的界(Seung，1995)，如下例子所示。

例11.3　Gauss分布变量　为了说明(11.86)所描述的界的用途，考虑一个具零均值且方差为1的Gauss分布变量。对这个特殊情况，$\langle\log(1+e^{z_j})\rangle$的精确值是0.806。在(11.86)所描述的界为$[e^{0.5\xi^2}+e^{0.5(1-\xi)^2}]$，在$\xi=0.5$时取得最小值0.818。这个界比$\xi=0$时的标准界0.974紧紧地接近真实值(Saul et al.,1996)。■

回到目前的问题，将式(11.85)和(11.86)代入式(11.83)，得到证据$\mathbf{X}_\alpha=\mathbf{x}_\alpha$的瞬时对数似然函数的一个下界如下：

$$\mathcal{L}(\mathbf{w}) \geqslant -\sum_{j\in\mathcal{H}}[\mu_j\log\mu_j + (1-\mu_j)\log(1-\mu_j)] + \sum_{j\in\mathcal{H}}\sum_{i<j} w_{ji}\mu_i(\mu_j-\xi_j) - \sum_{j\in\mathcal{H}}\log\langle\exp(-\xi_j z_j)+\exp((1-\xi_j)z_j)\rangle \tag{11.87}$$

其中z_j由式(11.84)定义。这是在一个一个例子的基础上计算对数似然函数$\mathcal{L}(\mathbf{w})$的一个理想的界。

sigmoid 信度网络平均场逼近的学习过程

在导出式(11.87)的界时我们引入了两组变分参数：$\mu_j(j\in\mathscr{H})$和ξ_j(对所有j)，但没有具体指定它们。这些都是可调参数，既然目标是最大对数似然函数$\mathscr{L}(\mathbf{w})$，我们自然选择μ_j和ξ_j的值使得它们最大化(11.87)的右边表达式。为了这一点我们使用 Saul et al.(1996)描述的两步迭代过程。 [582]

考虑第一种情形：均值μ_j固定，而要求寻找参数ξ_j的值使之产生对数似然函数$\mathscr{L}(\mathbf{w})$的最紧的界。这里我们注意，式(11.87)右边的表达式没有耦合属于网络不同神经元的ξ_j的项。因此，关于ξ_j求表达式的最小值归结为在[0，1]上求N个独立的最小值，这里N为网络神经元的总体数目。

考虑第二种情形：ξ_j的值固定，要求寻找均值μ_j使之产生对数似然函数$\mathscr{L}(\mathbf{w})$的最紧的界。为此我们引入下列定义：

$$K_{\mu} = -\frac{\partial}{\partial\mu_j}\log < \exp(-\xi_j z_j) + \exp((1-\xi_j)z_j) > \tag{11.88}$$

其中随机变量z_j由式(11.84)定义。给定证据(样本)$\mathbf{x}_\alpha\in\mathscr{T}$，偏导数$K_{\mu}$提供神经元$i$的状态$x_i$对神经元$j$的状态$x_j$的亲缘影响的一种度量。由 sigmoid 信度网络突触权值的定义，只有当状态x_i是状态x_j的双亲时K_{ji}才不为零。利用式(11.82)的析因分布，我们可以求得$(-\xi_j z_j)$和$\exp((1-\xi_j)z_j)$的整体均值，从而求出偏导数K_{μ}，这里计算K_{ji}的公式在表 11-5 给出。有了K_{μ}的值，我们可以继续对固定的ξ_j寻找参数μ_j值以最大化对数似然函数$\mathscr{L}(\mathbf{w})$的过程。特别，对μ_j求式(11.87)的微分，令微分值为 0，重新调整项后我们得到

$$\log\left(\frac{\mu_j}{1-\mu_j}\right) = \sum_{i<j}[w_{ji}\mu_i + w_{ij}(\mu_i - \xi_i) + K_{ij}]$$

可写成等价的形式

$$\mu_j = \varphi\left(\sum_{i<j}[w_{ji}\mu_i + w_{ij}(\mu_i - \xi_i) + K_{ij}]\right) \text{对 } j\in\mathscr{H} \tag{11.89}$$

其中$\varphi(\cdot)$是 sigmoid 函数。方程(11.89)称为 sigmoid 信度网络的*平均场方程*。在这个方程中 sigmoid 函数的变量构成神经元j的所谓 Markov *层*，它组成如下：

- 神经元j的双亲和孩子，分别由项$w_{ji}\mu_i$和$w_{ij}\mu_{ij}$表示。
- 神经元的孩子的其他双亲，通过偏导数K_{ij}继承。

图 11-9 表示神经元j的 Markov 层，“Markov 层”的思想由 Pearl(1988)引入；它说明神经元j的有效输入由它的双亲、孩子和孩子的双亲这些项组成。

虽然作为真实后验分布$P(\mathbf{X}_\beta=\mathbf{x}_\beta|\mathbf{X}_\alpha=\mathbf{x}_\alpha)$的一个逼近，在式(11.82)中选择析因分布并不精确，但是平均场方程(11.89)选择参数$\{\mu_j\}_{j\in\mathscr{H}}$的最优值使得这个逼近尽可能准确。这样依次在一个例子接一个例子的基础上计算对数似然函数$\mathscr{L}(\mathbf{w})$的最紧平均场界(Saul et al.，1996)。 [583]

在计算参数$\{\xi_j\}$和$\{\mu_j\}$的更新值后，接着计算突触权值w_{ji}的修正，使用公式

$$\Delta w_{ji} = \eta\frac{\partial B(\mathbf{w})}{\partial w_{ji}} \tag{11.90}$$

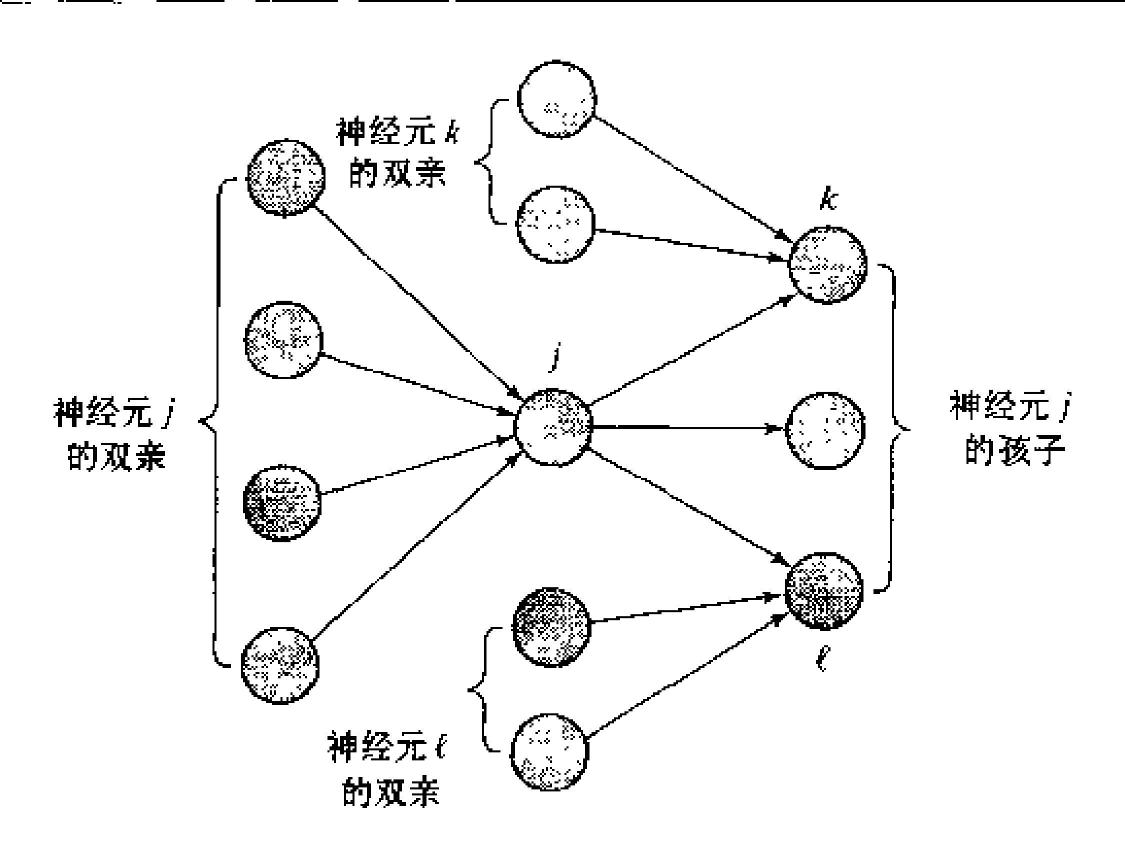


图 11-9 Markov 层举例

其中 η 是学习率参数，$B(\mathbf{w})$是对数似然函数$\mathscr{L}(\mathbf{w})$的下界；即 $B(\mathbf{w})$为式(11.83)右边的表达式。利用这个表达式，直接求取偏导数$\partial B(\mathbf{w})/\partial w_{ji}$的值。

表 11-3 给出 sigmoid 信度网络平均场逼近的学习过程的小结。这个表包括计算偏导数 K_{ji} 和$\partial B(\mathbf{w})/\partial w_{ji}$的公式。

表 11-3 用于 sigmoid 信度网络平均场逼近的学习过程

初始化。初始化网络权值 w_{ji}为$[-a, a]$内均匀分布的随机值，a 的典型值为 0.5。

计算。从训练集$\mathscr{T}$抽取样本 $\mathbf{x}_\alpha$，进行下列计算：

1. 对固定的$\{\mu_j\}$更新$\{\xi_j\}$。

固定后验分布 $P(\mathbf{X}_\beta = \mathbf{x}_\beta | \mathbf{X}_\alpha = \mathbf{x}_\alpha)$的析因逼近的均值$\{\mu_j\}_{j\in\mathscr{H}}$，最小化下列对数似然函数的界：

$$B(\mathbf{w}) = -\sum_{j\in\mathscr{H}}[\mu_j\log\mu_j + (1-\mu_j)\log(1-\mu_j)] + \sum_i\sum_{\substack{j\in\mathscr{H}\\ i<j}} w_{ji}\mu_i\mu_j - \sum_i\sum_{\substack{j\in\mathscr{H}\\ i<j}} w_{ji}\mu_i\xi_j - \sum_{j\in\mathscr{H}}\log\langle\exp(-\xi_j z_j) + \exp((1-\xi_j)z_j)\rangle$$

其中

$$z_j = \sum_{i<j} w_{ji}x_i$$

$B(\mathbf{w})$的最小化归结为在区间$[0, 1]$内 N 个独立最小化。

2. 对固定的$\{\xi_j\}$更新$\{\mu_j\}$

对固定参数值$\{\xi_j\}$，迭代平均场方程

$$\mu_j = \varphi\Big(\sum_{i<j}[w_{ji}\mu_i + w_{ij}(\mu_i - \xi_i) + K_{ij}]\Big)$$

其中

$$K_{ji} = -\frac{\partial}{\partial\mu_i}\log\langle\exp(-\xi_j z_j) + \exp((1-\xi_j)z_j)\rangle$$

$$= \frac{(1-\theta_j)(1-\exp(-\xi_j w_{ji}))}{1-\mu_i+\mu_i\exp(-\xi_j w_{ji})} + \frac{\theta_j(1-\exp((1-\xi_j)w_{ji}))}{1-\mu_i+\mu_i\exp((1-\xi_j)w_{ji})}$$

$$\theta_j = \frac{\langle\exp((1-\xi_j)z_j)\rangle}{\langle\exp(-\xi_j z_j) + \exp((1-\xi_j)z_j)\rangle}$$

$$z_j = \sum_{i<j} w_{ji}x_i$$

（续）

函数 $\varphi(\cdot)$ 为 sigmoid 函数

$$\varphi(v) = \frac{1}{1+\exp(-v)}$$

3. 突触权值修正，

对于参数 $\{\mu_j\}$ 和 $\{\xi_j\}$ 的更新值，计算突触权值 Δw_{ji} 的修正量

$$\Delta w_{ji} = \eta \frac{\partial B(\mathbf{w})}{\partial w_{ji}}$$

其中 η 是学习率参数，且

$$\frac{\partial B(\mathbf{w})}{\partial w_{ji}} = -(\xi_j - \mu_j)\mu_i + \frac{(1-\theta_j)\xi_j\mu_i\exp(-\xi_j w_{ji})}{1-\mu_i+\mu_i\exp(-\xi_j w_{ji})} - \frac{\theta_j(1-\xi_j)\mu_i\exp((1-\xi_j)w_{ji})}{1-\mu_i+\mu_i\exp((1-\xi_j)w_{ji})}$$

其中 θ_j 已定义。更新突触权值：

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

4. 对训练集 $\mathscr{T}$ 循环，

对包含在训练集的所有训练样本进行循环，从而最大化它们的似然函数到一个固定迭代次数，或者直到过拟合发生，例如用交叉验证方法检查出过拟合问题。

585

11.13 确定性退火

现在进入本章最后一个论题，确定性退火。在 11.5 节我们讨论模拟退火，这个随机松弛技巧提供解决非凸优化问题的一个强有力方法。但是必须仔细选择退火进度表。特别地，只有当退火温度的下降率不比对数更快时，全局最小才能得到保证。这种要求使得在许多应用中用模拟退火变得不现实。模拟退火的运行是在能量曲面(地形)上进行随机移动。相反，在确定性退火时，随机性以某种形式结合到能量或代价函数里，因此在一系列下降温度情况下进行确定性最优化(Rose et al.,1990;Rose,1998)；不要把确定性退火和平均场退火(这个术语常用来表示确定性 Boltzmann 机)混淆。

下面我们在无监督学习任务即聚类[11]的背景下，叙述确定性退火的思想。

通过确定性退火聚类

聚类定义为对一给定数据点集剖分成子集，使得每个子集尽可能是相似的。聚类是典型的非凸优化问题，因为实际上用于聚类的畸变函数都是输入数据的非凸函数。同时畸变函数关于输入的曲线充满局部最小，这使得求全局最小变得更为困难。

在 Rose(1991,1998)中通过剖分的随机化或等价的编码规则的随机化，对聚类描绘一个概率框架。这里利用的主要原则就是每个数据点以概率归为一特定聚类(子集)。具体地，令随机向量 $\mathbf{X}$ 表示源(输入)向量，令随机向量 $\mathbf{Y}$ 表示从感兴趣的码本的最优重构(输出)向量。这两个向量的单独实现分别记为 $\mathbf{x}$ 和 $\mathbf{y}$。

586

对聚类我们需要一个畸变度量，由 $d(\mathbf{x},\mathbf{y})$ 表示。假定 $d(\mathbf{x},\mathbf{y})$ 满足两个希望的性质：(1)对任何 $\mathbf{x}$ 它是 $\mathbf{y}$ 的凸函数，(2)当变元有限时，它是有限的。例如，Euclid 平方畸变度量

$$d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 \tag{11.91}$$

满足这种适度的假定。对随机模式的期望畸变定义为

$$D = \sum_{\mathbf{x}}\sum_{\mathbf{y}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})d(\mathbf{x},\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x}) \cdot \sum_{\mathbf{y}} P(\mathbf{Y}=\mathbf{y}\mid\mathbf{X}=\mathbf{x})d(\mathbf{x},\mathbf{y}) \tag{11.92}$$

其中 $P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})$是 $\mathbf{X}=\mathbf{x}$ 和 $\mathbf{Y}=\mathbf{y}$ 联合事件的概率。在式(11.92)的第二个等式，我们利用联合事件概率公式

$$P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y}) = P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x})P(\mathbf{X}=\mathbf{x}) \tag{11.93}$$

条件概率 $P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x})$指联想概率，即，码字向量 $\mathbf{y}$ 联想源向量 $\mathbf{x}$ 的概率。

传统上通过对聚类模型的自由参数，即重建向量 $\mathbf{y}$ 和联想概率 $P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x})$，最小化期望畸变 D。这种形式的最小化产生"硬"聚类解，硬是指源向量 $\mathbf{x}$ 被归入最近的码向量 $\mathbf{y}$。另一方面，在确定性退火中，优化问题被改变成寻找服从特定随机水平概率分布，使得它最小化期望畸变。作为随机水平的一个主要度量，我们使用 Shannon 熵，定义为(参看 10.4 节)

$$H(\mathbf{X},\mathbf{Y}) = -\sum_{\mathbf{x}}\sum_{\mathbf{y}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})\log P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y}) \tag{11.94}$$

期望畸变的约束优化可以表示成 Lagrange 函数

$$F = D - TH \tag{11.95}$$

的最小化，其中 T 为 Lagrange 乘子。从式(11.95)我们观察到：

- 对大的 T 值，熵 H 被最大化。
- 对小的 T 值，期望畸变 D 被最小化，导致硬(非随机)聚类解。
- 对中间的 T 值，F 的最小值提供在熵 H 增加和期望畸变 D 减少之间的折中。

587　最为重要的，比较式(11.11)和式(11.95)，我们可以确认表 11-4 所列的约束聚类优化问题和统计力学之间的对应。根据这种类比，我们今后称 T 为温度。

表 11-4　约束聚类和统计物理学之间的对应

约束聚类优化	统计物理学
Lagrange 函数 F	自由能量 F
期望畸变 D	平均能量 $<E>$
Shannon 熵 H	熵 H
Lagrange 乘子 T	温度 T

为了进一步了解 Lagrange 函数 F，我们注意联合熵 $H(\mathbf{X},\mathbf{Y})$可以分成如下两项(参看式(10.25))：

$$H(\mathbf{X},\mathbf{Y}) = H(\mathbf{X}) + H(\mathbf{Y} \mid \mathbf{X})$$

其中 $H(\mathbf{X})$为信源熵，$H(\mathbf{Y} \mid \mathbf{X})$为在给定源向量 $\mathbf{X}$ 后重建向量 $\mathbf{Y}$ 的条件熵。信源熵 $H(\mathbf{X})$是独立于聚类的。因此，我们可以从 Lagrange 函数 F 中去掉信源熵 $H(\mathbf{X})$，从而集中在条件熵

$$H(\mathbf{Y} \mid \mathbf{X}) = -\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x}) \sum_{\mathbf{y}} P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x})\log P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x}) \tag{11.96}$$

这样突出联想概率 $P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x})$的作用。因此，考虑到约束聚类优化问题和统计物理学之间的对应以及 11.2 节描述的最小自由能量原理，我们发现关于联想概率的 Lagrange 函数 F 的最小化导致 Gibbs 分布

$$P(\mathbf{Y}=\mathbf{y} \mid \mathbf{X}=\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}}\exp\left(-\frac{d(\mathbf{x},\mathbf{y})}{T}\right) \tag{11.97}$$

其中 $Z_{\mathbf{x}}$ 为当前问题的剖分函数，定义为

$$Z_{\mathbf{x}} = \sum_{\mathbf{y}}\exp\left(-\frac{d(\mathbf{x},\mathbf{y})}{T}\right) \tag{11.98}$$

当温度 T 接近无穷时，我们从式(11.97)发现联想概率趋向于均匀分布。这就意味着当温度

相当高时，每个输入向量是相等地联想起所有聚类。这种联想可以被视作“极度模糊”。在另一个极端，当温度 T 趋于零时，联想概率趋近于 δ 函数。因此，当温度较低，分类是“硬”的，每个输入样本以概率 1 分给最近的码向量。为了寻找 Lagrange 函数 F 的最小值，我们将式(11.97)的 Gibbs 分布代入式(11.92)和式(11.96)，然后将结果表达式用到式(11.95)的 588 Lagrange 算子 F 的公式中。这样做导致的结果为(参看习题 11.22)

$$F^* = \min_{P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})} F = -T\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x})\log Z_{\mathbf{x}} \tag{11.99}$$

对剩下的自由参数即码向量 $\mathbf{y}$，最小化 Lagrange 函数，我们置 F^* 关于 $\mathbf{y}$ 的梯度为零。因此，得到条件

$$\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})\frac{\partial}{\partial \mathbf{y}}d(\mathbf{x},\mathbf{y}) = \mathbf{0} \quad 对所有\ \mathbf{y}\in\mathscr{G} \tag{11.100}$$

其中 $\mathscr{G}$ 为所有码向量的集合。利用式(11.93)的公式和对 $P(\mathbf{X}=\mathbf{x})$ 规整化，可以重新定义这个最小化条件为

$$\frac{1}{N}\sum_{\mathbf{x}} P(\mathbf{Y}=\mathbf{y}\mid\mathbf{X}=\mathbf{x})\frac{\partial}{\partial \mathbf{y}}d(\mathbf{x},\mathbf{y}) = 0 \quad 对所有\ \mathbf{y}\in\mathscr{G} \tag{11.101}$$

其中联想概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$ 由式(11.97)的 Gibbs 分布定义。在式(11.101)中仅为了完整性包括比例因子 $1/N$，这里 N 为可用样本的数目。

我们现在可以描述聚类的确定性退火算法(Rose, 1998)。基本上，算法由以下两步组成：开始在温度 T 为很高值时对码向量最小化 Lagrange 函数 F^*，然后在降低温度 T 的同时跟踪最小值。换句话说，确定性退火运行时具有特定的退火进度表，温度依次降低。对温度 T 的每一值，执行算法核心的两步迭代可描述如下：

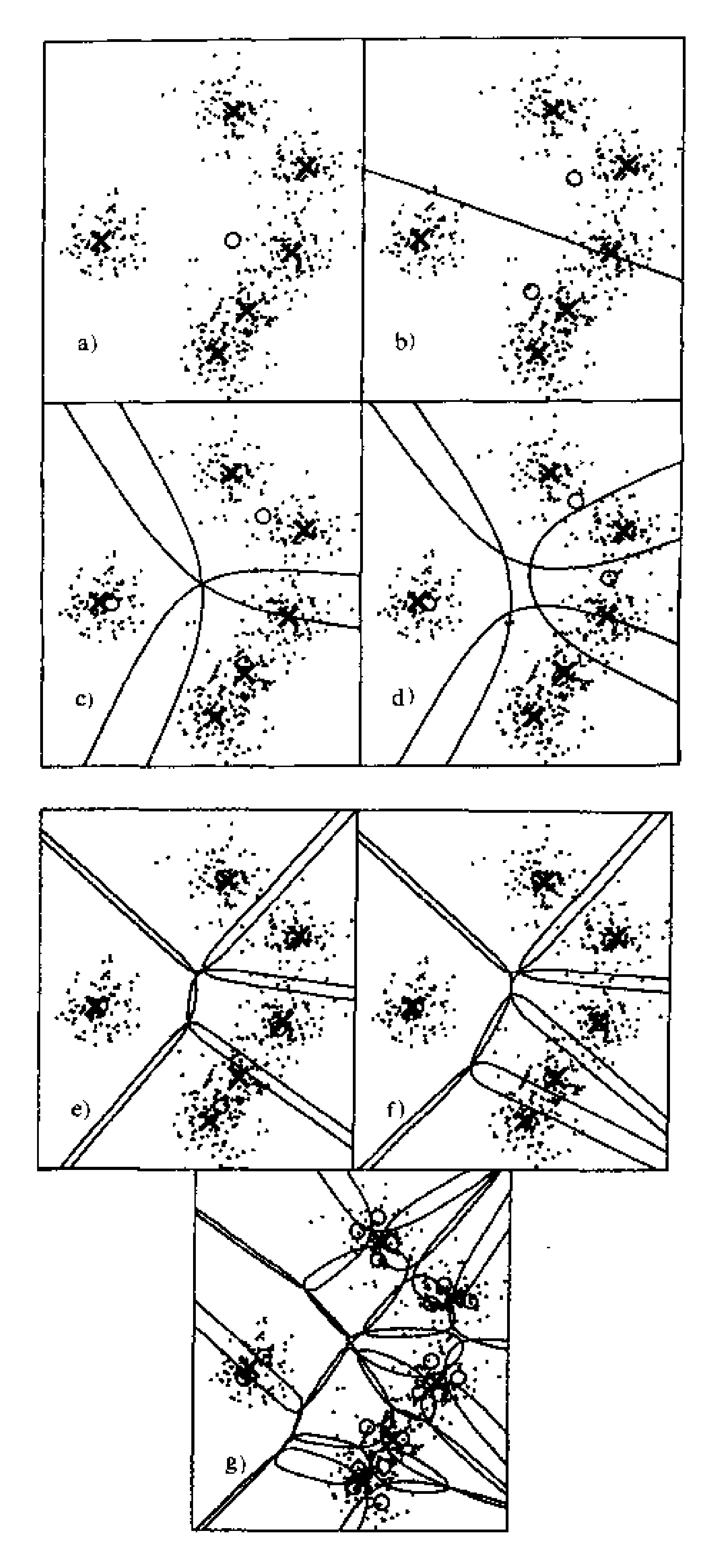
1. 固定码向量，利用对于给定畸变度量 $d(\mathbf{x},\mathbf{y})$ 的式(11.97)的 Gibbs 分布计算联想概率。
2. 固定联想，使用式(11.101)对码向量 $\mathbf{y}$ 最优化畸变度量 $d(\mathbf{x},\mathbf{y})$。

这个两步迭代过程对 F^* 单调不升，因此能保证收敛到一个最小点。当温度 T 很高时，Lagrange 算子 F^* 相当光滑，而且在前面对畸变度量 $d(\mathbf{x},\mathbf{y})$ 的适度假设下，F^* 是 y 的凸函数。在温度较高时可以求得 F^* 的全局极小。随着温度降低，联想概率变“硬”，导致一个“硬”的聚类解。

当温度 T 按退火进度表降低，系统经历一系列相变，相变由自然聚类分叉组成，在分叉处聚类模型规模(即聚类的数目)增加(Rose et al., 1990; Rose, 1991)。这种现象由于以下原因而富有意义：

- 它提供控制聚类模型大小的一个有用工具。
- 正如通常的物理退火一样，相变是确定性退火的关键点，此处需要小心进行退火。 589
- 关键点是可计算的，因而提供用于在两个相变之间加速算法的信息。
- 最优模型大小可以确认，通过耦合一个确认过程检验在不同相位得到的一系列解，这些解是表示模型规模(即聚类的数目)逐渐升高的解。

例 11.4 图 11-10 和图 11-11 举例说明随温度 T 下降或温度倒数 $B=1/T$ 的上升，确定性退火在不同相位时聚类解的演化，产生这些图所使用的数据集由 6 个 Gauss 分布混合而成，它们的中心都以 X 标识、计算所得聚类的中心都以 o 标识。由于聚类解在非零温度不是“硬”分类的，这个随机划分在图中由属于该聚类的等概率——如概率为 1/3 的围线所描绘。



590

图 11-10 不同相位的聚类，画线是等概率围线，在 b)中 $p=1/2$，其余情况下 $p=1/3$

a)1 个聚类($B=0$) b)2 个聚类($B=0.0049$)

c)3 个聚类($B=0.0056$) d)4 个聚类($B=0.0100$)

e)5 个聚类($B=0.0156$) f)6 个聚类($B=0.0347$)

g)19 个聚类($B=0.0605$)

这个过程开始只有一个自然聚类(图 11-10a)包括所有训练集。在第一次相变，它分裂成两个聚类(图 11-10b)，然后经过一系列相变直到它达到 6 个聚类的自然集。当所有聚类都分裂时，下一个相变导致"爆炸"。图 11-11 表示相位图，显示随退火过程的进行平均畸变变量变化的情况，以及在每个相阶段，自然聚类的数目。在这个图中，平均畸变(相对它的最小值规整化)是对温度 T 的倒数即 B(相对于它的最小值规整化)画出的。两个坐标轴都是以它们相关的对数形式标出的。

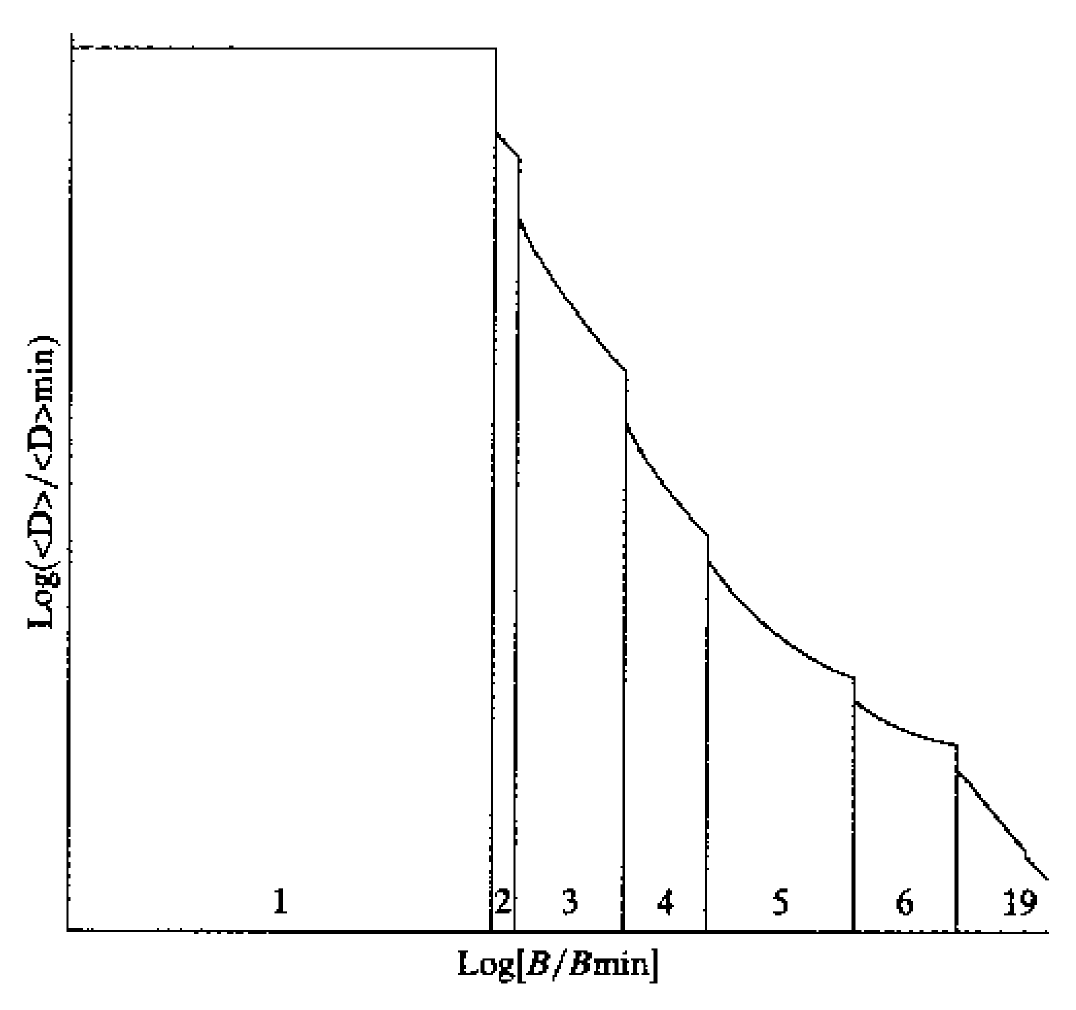


图 11-11 混合 Gauss 分布样本的相位图。对每个相位显示有效聚类的数目

和 EM 算法的类比

为了说明确定性退火算法的另一个重要方面，假设我们将联想概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$看成一个二值随机变量 V_{xy}的期望值，其中 V_{xy}定义为

$$V_{xy} = \begin{cases} 1 & \text{若源向量 } \mathbf{x} \text{ 被归入码向量 } \mathbf{y} \\ 0 & \text{否则} \end{cases} \tag{11.102}$$

从这个观点出发，我们认识到确定性退火算法的两步迭代是第 7 章描述的用于最大似然估计的期望最大(EM)算法的一种形式。特别在第一步中计算联想概率，我们有与它等价的求期望步骤。在第二步最小化 Lagrange 函数 F^*，我们有与它等价的最大化步骤。

但在作这种类比时，注意确定性退火比最大似然估计是更一般的。我们这样说，是因为与最大似然估计不一样，确定性退火不对数据的固有概率分布作任何假定。事实上，联想概率是由最小化 Lagrnage 函数导出的。

11.14 小结和讨论

在这一章中我们讨论利用植根于统计力学的思想作为优化技术表示和学习机器的数学基础。这里考虑的学习机器可分类如下：

- 随机机器，例如 Boltzmann 机、sigmoid 信度网络和 Helmholtz 机。
- 确定性机器，利用平均场逼近从 Boltzmann 机和 sigmoid 信度网络导出。

Boltzmann 机使用隐藏的和可见的随机二值状态的神经单元，它巧妙地利用 Gibbs 分布的良好性质，从而具有一些吸引人的特征：

- 通过训练，神经元所显示的概率分布和环境相匹配。
- 网络提供一种推广的方法，可用于搜索、表示和学习的基本问题(Hinton,1989)。
- 592 如果退火进度表在学习过程中足够慢，则网络保证找到状态能量曲面的全局最小值(Geman and Geman,1984)。

遗憾的是退火进度表太慢以至没有实用价值。但是，针对具体的 Boltzmann 机学习过程可以进行加速，对这些 Boltzmann 机我们无需进行采样算法或者应用平均场逼近。特别地，如果 Boltzmann 机隐藏神经元是链状或树状以及它们的耦合对，学习可在多项式时间完成。之所以能取得这样的结果，是因为应用了统计力学中熟知的"抽取"算法，它是一个简单而精致的过程，非常像求解电阻电感电容(RLC)电路一样，从图中递归地删除连接和节点(Saul and Jordan,1995,1996)。

sigmoid 信度网络给出 Boltzmann 机的一个重要改进，它消除学习过程中的负向(自由运行)阶段。这是由于它们不用 Boltzmann 机中对称连接而使用有向无圈连接。也就是说 Boltzmann 机是一种具有反馈的递归网络，而 sigmoid 信度网络是无反馈的多层结构。正如名字所暗示的，sigmoid 信度网络和由 Pearl(1988)首先提出的经典的信度网络非常接近，因而将神经网络的研究和概率推理模型及图形模型联系起来(Jordan,1998; Jordan et al.,1998)。

Helmholtz 机又与它们不同。它的发展受到视觉是图形取反(Horn,1997; Hinton & Ghahramani,1997)的思想的启迪。特别在反向运行中它使用一个随机的产生模型把一个场景的抽象表示转化为一个深度图像。场景的抽象表示(即网络自己关于世界的视觉知识)是由于前向运行的随机识别模型学习的。通过识别模型和产生模型的巧妙结合(即前向/反向投影)，Helmholtz 机起到自监督机的作用，因而不需要教师。

接着讨论确定性机器，确定性 Boltzmann 机是由 Boltzmann 机导出的，它用两个随机变量均值的乘积替代两个随机变量乘积的均值，这是平均场逼近的朴素形式。这样做使得确定性 Boltzmann 机比标准的随机 Boltzmann 机快许多。遗憾的是在实际应用中严格限制在仅含一个隐藏层的情况。在 Kappen and Rodriguez(1998)中，讨论到在对 Boltzmann 机正确应用平均场理论时，使用线性响应定理计算相关性。这个定理的本质在于应用其线性响应的逼近替代式(11.53)的 Boltzmann 学习规则中箝制和自由运行时的相关性。根据 Kappen and Rodriguez(1998)的讨论，新的学习过程可应用于含有或没有隐藏神经元的网络。

sigmoid 信度网络的确定性形式的导出是应用平均场理论的另一个形式，应用 Jensen 不等式导出对数似然函数的一个严格下界。进一步，以一种原则化的方式利用易处理的子结构优点，理论上使得这类神经网络成为信度网络之外的另一种重要网络类型。

在本章我们还讨论两个优化技术：模拟退火和确定性退火。模拟退火的突出点在于在能593量曲面上进行随机移动，从而使得退火进度表非常慢，这样使得在许多应用中无法实际使用。相反，确定性退火将随机性耦合到代价函数中，从一个较高温度开始，然后逐渐降低，在每个依次的温度对目标函数进行确定性的优化。但是，注意模拟退火保证到达全局极小，而确定性退火还没有找到这种保证。

本章中我们虽然强调应用优化技术和随机机器解决无监督学习任务，但如果需要也可以应用于监督学习任务。

注释和参考文献

[1] 在 11.3 中描述的术语"典型分布"是由 J.Willard Gibbs(1902)在《统计力学的基本原理》第一部分 33 页上创造的新名词，他写到

"所表示的分布……

$$P = \exp\left(\frac{\Psi - \varepsilon}{H}\right)$$

看来代表了最简单可以想象的情况，因为当系统包括分离能量的部分时，它的分布和分离部分的相位的分布律相同，其中 H 和 Ψ 为常数，且 H 为正。分布的这个性质极大地简化了讨论，是和热力学极端重要关系的基础。当一个整体系统在相位以刚才描述的方式分布，即当概率(P)指标是能量(ε)的线性函数，我们将说整体是*典型分布的*，称能量的除数 H 为分布的模。"

在物理文献中，式(11.3)通常称为*典型分布*(Reif, 1965)或 Gibbs 分布(Landau and Lifschitz,1980)。在神经网络文献中称为 Gibbs 分布、Boltzmann 分布和 Boltzmann-Gibbs 分布。

[2] 引入温度和模拟退火到组合优化问题的想法是由 Kirkpatrick, Gelatt and Vacchi(1983)三人和 Cerny(1985)独立提出的。

在物理环境中，退火是自然界的一个精细的过程。Kirkpatrick 等在 1983 的文章中讨论"熔化"一个固体的概念，这涉及升高温度到一个最大值使得固体的所有粒子处于液态时能够随机地运动。接着降低温度，使得所有粒子调整到具有低能基态的相应格点。如果冷却太快，也就是说，在每一温度，固体没有足够时间达到热平衡，这样得到的晶体会有许多缺陷，或物质将形成无晶体序的玻璃体并且仅为局部最优结构的亚稳态。"熔化"这个概念对于思考玻璃体可能是正确的方法，或许对考虑组合优化问题的计算也有帮助。但是当讨论许多其他应用领域时会失误(Beckerman,1997)。例如，在图像处理中，如果我们升高温度使得所有粒子能够随机地调整自己的位置，就会丢失图像——变成均匀灰度。在相应的冶金学意义上，当退火铁或铜时，我们必须保证退火温度低于熔点；否则将会毁坏样本。

有几个控制冶金退火重要的参数：

- *退火温度*，指示金属或合金加热到什么温度。
- *退火时间*，指定保持提高温度后的时间长度。
- *退火进度表*，指定温度下降的速度。

594

在描述退火进度表的小节中可以发现，这些参数在模拟退火里能找到和它们相对应的部分。

[3] Langevin 方程(具有时变温度)提供了另一个由 Grenander(1983)提出的全局最优化算法的基础，随后由 Gidas(1985)进行了分析。Langevin 方程是随机微分方程，描述为(Reif, 1965)

$$\frac{dv(t)}{dt} = -\gamma v(t) + \Gamma(t)$$

其中 $v(t)$ 为浸入粘性流中质量为 m 的粒子的速度，γ 为常数，等于磨擦系数和质量 m 的比值，$\Gamma(t)$ 为每单位质量的波动力。Langevin 方程是描述非平衡热动力学的第一个数学方程。

[4] 对更复杂的和理论上的退火进度表，参看书籍 Aarts and Korst(1989, pp.60 - 75)和 van Laarhoven and Aarts(1988, pp.62 - 71)。

[5] Gibbs 抽样在统计物理中称为 Metropolis 算法的"热浴"形式。自从在 Geman and Geman (1984)及 Gelfand and Smith(1990)的文献中正式出现以后，它被广泛应用于图像处理、神经网络和统计学。后一篇文章还讨论抽样(或 Monte Carlo)的其他方法，这些方法基于对边缘概率估计的数值计算。Hastings(1970)给出了 Metropolis 算法的推广，而 Gibbs 抽样仅是它的特例，提到了它在解决统计中数值问题的潜在应用。

[6] Boltzmann 机的可见神经元可以被分成输入和输出神经元。在第二种结构中 Boltzmann 机是在教师监督下进行联想，输入神经元从环境接受信息而输出神经元报告计算结果给最终用户。

[7] 式(11.39)的表达式适合于 Boltzmann 机的"开"和"关"状态分别用 +1 和 -1 表示。如果机器利用 1 和 0 分别表示"开"和"关"状态，我们有

$$E(\mathbf{x}) = -\sum_{i}\sum_{\substack{j \\ i\neq j}} w_{ji}x_i x_j$$

[8] 传统上，相对熵或 Kullback - Leibler 散度用作 Boltzmann 机的性能指标(Ackley et al., 1985; Hinton and Sejnowski, 1986)。这个标准提供环境和物理内部模型之间的差异的度量，定义为

595

$$D_{p_\alpha^+\|p_\alpha^-} = \sum_\alpha p_\alpha^+ \log\left(\frac{p_\alpha^+}{p_\alpha^-}\right)$$

其中 p_α^+ 为网络被箝制时可见神经元在状态 α 的概率，p_α^- 为网络自由运行时可见神经元在状态 α 的概率。网络突触权值被调整，使 $D_{p_\alpha^+\|p_\alpha}$ 达到最小；参看习题 11.10。

当应用于训练集时，最小化 Kullback - Leibler 散度原则和最大似然原则基本上等价。为了看清这个等价性，我们注意两个分布 f 和 g 的 Kullback - Leibler 之间散度由

$$D_{f\|g} = -H(f) - \sum f\log(g)$$

给出。如果分布 f 由训练集确定，给定 g 的一个优化模型，第一项是常数，第二项则是负的对数似然函数。因此最小化 Kullback - Leibler 散度是和最大似然等价的。

[9] 信度网络最初是为了表示专家系统中的概率知识而引入的。在文献中它们也指 Bayes 网络。

[10] Helmholtz 机属于以前向投影和反向投影为特征的一类神经网络。前向投影的思想起源于 Grossberg(1980)的自适应共振理论研究；也可参看 Carpenter and Grossberg(1987)。在这个模型中，前向自适应滤波结合反向模板匹配，使得产生自适应共振(即放大和延长神经活动)。与 Grossberg 的自适应共振理论相反，对于试图准确捕获输入数据的固有结构的产生模型，Helmholtz 机利用统计方法把自监督学习作为一种确定产生模型的方法。另一个紧密相关的工作是 Luttrell(1994, 1997)的工作。在 Luttrell(1994)的工作中，提出了"折叠 Markov 链"(folded Markov chain, FMC)的思想。特别，一个 Markov 链前向转移之

后，紧接着利用同样的链按反向进行逆转移(利用 Bayes 定理)。在 Luttrell(1997)中，讨论 FMC 和 Helmholtz 机的关系。

另外一些相关工作包括诸如 Kawato et al.(1993)的工作，其中考虑以与 Helmholtz 机相似但没有概率关系的方式前向(识别)模型和反向(产生)模型，以及 Mumford(1994)关于映射 Grenander 产生模型到人脑中的提议。

在 Dayan and Hinton(1996)中，提及大量不同种类的包括监督方法的 Helmholtz 机。

[11] 确定性退火已成功应用到许多学习任务：

- 向量量化(Rose et al.,1992;Miller and Rose,1994)
- 统计分类设计(Miller et al.,1996)
- 利用混合专家的非线性回归(Rao et al.,1997a)
- 隐藏 Markov 模型的语音识别(Rao et al.,1997b)

一个隐 Markov 模型类似于 Markov 链，因为它们从一个状态转移到另一个状态都是根据概率的。但它们有一个重要区别，在 Markov 链中，输出符号的产生是确定的。另一方面，在隐 Markov 模型中，输出符号是概率性的，这样所有符号都可能达到每一状态。因此对隐 Markov 模型的每一状态，我们有所有输出符号的概率分布。隐 Markov 模型的讨论可参见 Rabiner(1989)，Rabiner and Juang(1986) 和 Jelinek(1997)。 596

习题

Markov 链

11.1 从状态 i 到状态 j 的 n 步转移概率记为 $p_{ij}^{(n)}$。利用归纳法证明

$$p_{ij}^{(1+n)} = \sum_k p_{ik} p_{kj}^{(n)}$$

11.2 图 11-12 表示随机行走过程的状态转移图，其中转移概率 p 大于零。图中所示的无限长 Markov 链是不可约吗？说明你回答的理由。

11.3 考虑图 11-13 所描绘 Markov 链，它是可约的。找出包含在这个状态转换图中的各个状态类。

11.4 计算图 11-14 所示的 Markov 链的稳定态的概率。

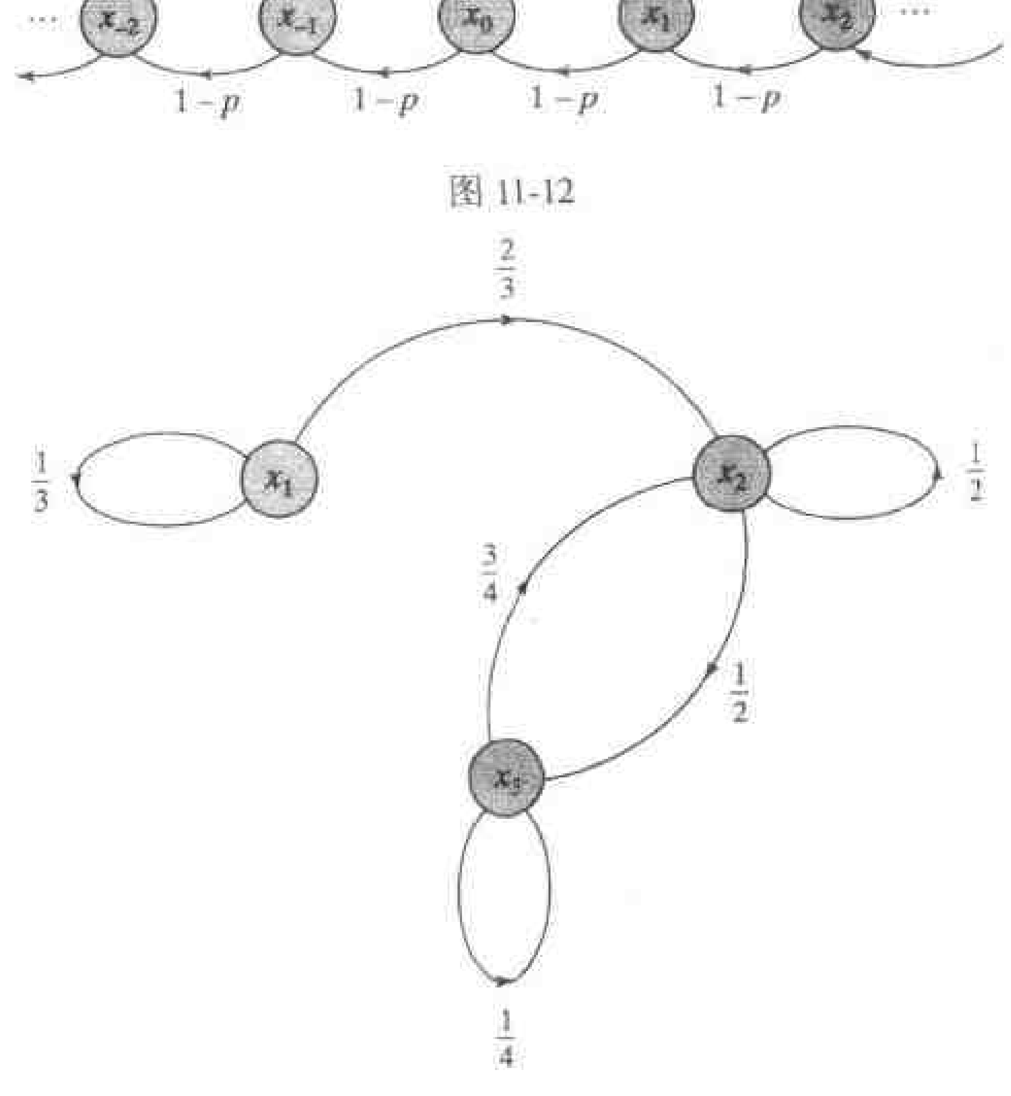


图 11-12

图 11-13

模拟技术

11.5 Metropolis 算法和 Gibbs 抽样器代表两类不同的模拟大规模问题的技术。讨论它们之间的基本相似点和不同点。

11.6 本题中考虑用模拟退火求解旅行商问题(traveling salesman problem, TSP)。条件如下:

- N 个城市
- 每两个城市间距离为 d
- 旅行路线为一个闭合的路径，只访问每个城市一次

597

目标是寻找具有最小总长度 L 的旅行路线(即排列城市访问的顺序)。在这个习题中，不同的可能旅行路线称为构形，而需最小化的代价函数为旅行路线的总长度。

(a)设计出一种产生合法构形的迭代方法。

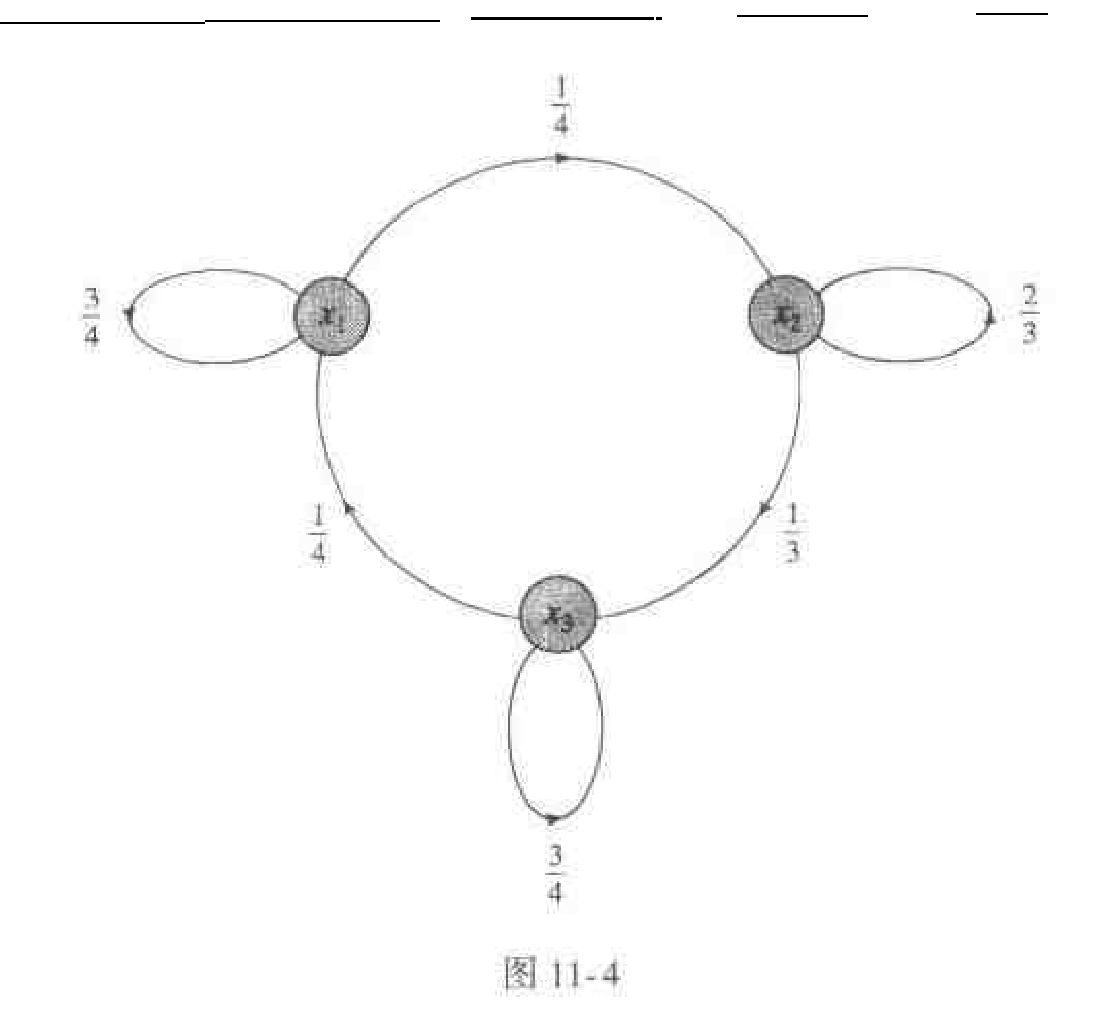


图 11-4

(b)旅行路线总长度定义为

$$L_P = \sum_{i=1}^{N} d_{P(i)P(i+1)}$$

其中 P 表示一个置换且 $P(N+1)=P(1)$。因此，剖分函数为

$$Z = \sum_P e^{-L_P/T}$$

其中 T 为控制参数。建立用于 TSP 的模拟退火算法。

Boltzmann 机

11.7 考虑一个在温度 T 运行的随机二值神经元 j。它从状态 x_j 翻转到状态 $-x_j$ 的概率为

$$P(x_j \rightarrow -x_j) = \frac{1}{1+\exp(-\Delta E_j/T)}$$

其中 ΔE_j 为翻转所导致的能量改变。Boltzmann 机的总能量定义为

598

$$E = -\frac{1}{2}\sum_{i}\sum_{\substack{j \\ i \neq j}} w_{ji}x_i x_j$$

其中 w_{ji} 为从神经元 i 到神经元 j 的突触权值，且 $w_{ji}=w_{ij}$ 和 $w_{ii}=0$。

(a)证明 $\Delta E_j = -2x_j v_j$，其中 v_j 为神经元 j 的诱导局部域。

(b)因此，证明神经元 j 从初态 $x_j=-1$ 翻转到 $x_j=+1$ 的概率为 $1/(1+\exp(-2v_j/T))$。

(c)证明当神经元 j 从初态为 $+1$ 翻转到状态 -1 时(b)中的公式仍然正确。

11.8 推导式(11.49)中对数似然函数 $L(\mathbf{w})$ 关于 Boltzmann 机突触权值 w_{ji} 的导数公式。

11.9 Gibbs 分布可以利用自完备的数学方法推导出，而不依赖于统计物理的概念。特别地，一个两步 Markov 链模型的随机机器可用来导出形成 Boltzmann 机特殊性质的假设(Mazaika, 1987)。这一点也不令人惊奇，因为作为 Boltzmann 机运行的模拟退火本身具有 Markov 性质(van Laarhoven and Aarts, 1988)。

考虑在一个随机机器中神经元的状态转移模型由两个随机过程组成:

- 第一个过程决定尝试哪个状态转移。
- 第二个过程决定这次转移是否成功。

(a)表示状态转移概率 p_{ji} 为两个因子的乘积，即

$$p_{ji} = \tau_{ji}q_{ji} \qquad \text{对 } j \neq i$$

证明 $p_{ii} = 1 - \sum_{j \neq i}\tau_{ij}q_{ij}$。

(b)假设尝试率矩阵是对称的，

$$\tau_{ji} = \tau_{ij}$$

并且假设尝试成功的概率满足互补条件转移概率的性质，

$$q_{ji} = 1 - q_{ij}$$

由两个假设证明 $\sum_{i}\tau_{ji}(q_{ij}\pi_j + q_{ij}\pi_i - \pi_j) = 0$。

(c)假定 $\tau_{ji} \neq 0$，利用问题(a)中的结果证明 $q_{ij} = \dfrac{1}{1 + (\pi_i/\pi_j)}$。

(d)最后，进行变量变换：$E_i = -T\log\pi_i + T^*$，其中 T 和 T^* 为任意常数。由此推导：(i)$\pi_i = \dfrac{1}{Z}\exp\left(-\dfrac{E_i}{T}\right)$。(ii)$Z = \sum_{j}\exp\left(-\dfrac{E_j}{T}\right)$。(iii)$q_{ji} = \left(\dfrac{1}{1+\exp(-\Delta E/T)}\right)$，其中 $\Delta E = E_j - E_i$。 599

(e)你能从这些结果中得出什么结论？

11.10 在 11.7 节我们利用最大似然函数作为推导式(11.53)所描述的 Boltzmann 学习规则的准则。在这个习题中我们利用别的准则重新考虑这个学习规则。由第 10 章的讨论，两个概率 p_α^+ 和 p_α^- 的 Kullback-Leibler 散度定义为

$$D_{p^+ \| p^-} = \sum_{\alpha} p_\alpha^+ \log\left(\frac{p_\alpha^+}{p_\alpha^-}\right)$$

其中对所有可能的状态 α 求和。概率 p_α^+ 表示网络在箝制(正向)状态时可见神经元处于状态 α 的概率，概率 p_α^- 表示网络在自由运行(负向)状态时可见神经元处于状态 α 的概率。利用 $D_{p_\alpha^+ \| p_\alpha^-}$ 重新推导 Boltzmann 学习规则。

11.11 考虑 Boltzmann 机的可见神经元分成输入神经元和输出神经元。这些神经元的状态分别表示为 α 和 γ。隐藏神经元状态记为 β。这个机器的 Kullback-Leibler 散度定义为

$$D_{p_\alpha^+ \| p_\alpha^-} = \sum_{\alpha} p_\alpha^+ \sum_{\gamma} p_{\gamma|\alpha}^+ \log\left(\frac{p_{\gamma|\alpha}^+}{p_{\gamma|\alpha}^-}\right)$$

其中 p_α^+ 为输入神经元在状态 α 的概率，$p_{\gamma|\alpha}^+$ 为给定输入状态 α 输出神经元被箝制在状态 γ 的条件概率，$p_{\gamma|\alpha}^-$ 为仅输入神经被箝制在状态 α 时处于热平衡中的输出神经元状态为 γ 的条件概率。和前面的一样，加号和减号上标分别表示正向(箝制)和负向(自由运行)条件。

(a)对输入、隐藏和输出神经元的 Boltzmann 机导出公式 $D_{p_\alpha^+ \| p_\alpha^-}$。

(b)对于这种网络配置经过重新解释相关性 ρ_{ji}^+ 和 ρ_{ji}^-，证明调整突触权值 w_{ji} 的 Boltzmann 学习规则仍可以被表示成和式(11.53)同样的形式。

sigmoid 信度网络

11.12 概述 Boltzmann 机和 sigmoid 信度网络之间的相似性和差异。

11.13 在习题 11.9 中，我们阐明了 Boltzmann 机可描述为两步 Markov 链模型。sigmoid

信度网络是否可以描述为一个 Markov 链模型？说明你的结论的理由。

11.14 令 w'_{ji} 表示 sigmoid 中从神经元 i 到神经元 j 的突触权值，用 +1 和 -1 分别表示“开”和“关”状态。如果 sigmoid 信度网络利用 1 和 0 表示神经元开状态和关状态，则令 w_{ji} 表示相应的突触权值，证明使用下面的变换 w_{ji} 可以转化成 w'_{ji}：

600

$$w'_{ji} = \frac{w_{ji}}{2} \qquad 对 0 < i < j$$

$$w'_{j0} = w_{j0} + \frac{1}{2}\sum_{0<i<j} w_{ji}$$

最后一行指的是神经元 j 的偏置值。

11.15 在 sigmoid 信度网络中我们确认概率 $P(\mathbf{X}_\beta = \mathbf{x}_\beta | \mathbf{X}_\alpha = \mathbf{x}_\alpha)$ 为 Gibbs 分布，概率 $P(\mathbf{X}_\alpha = \mathbf{x}_\alpha)$ 为相应的剖分函数。验证这两个结论的正确性。

Helmholtz 机

11.16 Helmholtz 机在识别模型和产生模型中都没有反馈。如果两个模型的任何一个中允许使用反馈，则网络的运行会怎样？

确定性 Boltzmann 机

11.17 如同习题 11.10 中所讨论，Boltzmann 机在概率空间作梯度下降(关于权值空间)。确定性 Boltzmann 机对什么函数作梯度下降？你可参考 Hinton(1989)讨论这个问题。

11.18 考虑具有非对称权值 $w_{ji} \neq w_{ij}$ 的递归网络。如果每次权值更新后它的长度向零衰减一个很小的比例，讨论确定性 Boltzmann 机学习算法将如何自动使网络成为对称的(Hinton，1989)。

确定性 sigmoid 信度网络

11.19 证明式(11.77)左边和右边表达式的差等于分布 $Q(\mathbf{X}_\beta = \mathbf{x}_\beta | \mathbf{X}_\alpha = \mathbf{x}_\alpha)$ 和 $P(\mathbf{X}_\beta = \mathbf{x}_\beta | \mathbf{X}_\alpha = \mathbf{x}_\alpha)$ 之间的 Kullback-Leibler 散度。

11.20 在式(11.89)中的 sigmoid 函数的变量定义确定性 sigmoid 信度网络中神经元 j 的诱导局部域 v_j，它和用反向传播算法训练的多层感知器中神经元相应的诱导局部域有什么差异？

确定性退火

11.21 在 11.13 节中我们利用信息论方法讨论确定性退火的思想。确定性退火的思想也可以基于第 10 章讨论的最大熵原理用原理化的方式产生。说明第二种方法的基本原理(Rose，1989)。

11.22 (a)利用式(11.97)和式(11.98)，推导式(11.99)所给出的 Lagrange 函数 F^* 的结果，该结果是用联想概率的 Gibbs 分布得到的。

(b)利用本题中(a)的结果，导出式(11.101)给出的 F^* 关于码向量 $\mathbf{y}$ 取最小值的条件。

(c)应用式(11.101)的最小化条件到式(11.91)的平方畸变度量，评论你的结果。

11.23 考虑数据集为混合 Gauss 分布，在这种情况下，怎样才能使得利用确定性退火比利用最大似然估计有优越性？

601

11.24 在本题中我们探讨基于神经网络的模型分类中确定性退火的应用(Miller et al.，

1996)。输出层的神经元 j 的输出记为 $F_j(\mathbf{x})$，其中 $\mathbf{x}$ 为输入向量。分类决策是基于最大判别式 $F_j(\mathbf{x})$。

(a)对于概率目标函数，考虑

$$F = \frac{1}{N}\sum_{(\mathbf{x},\mathscr{C})\in\mathscr{T}}\sum_{j}P(\mathbf{x}\in\mathscr{R}_j)F_j(\mathbf{x})$$

其中$\mathscr{T}$为带标号向量的训练集，$\mathbf{x}$ 表示输入向量，$\mathscr{C}$为它的类别标识，$P(\mathbf{x}\in\mathscr{R}_j)$为输入向量 $\mathbf{x}$ 和类别区域$\mathscr{R}_j$ 的联想概率。利用第 10 章讨论的最大熵原理，写出 $P(\mathbf{x}\in\mathscr{R}_j)$的 Gibbs 分布。

(b)令 $<P_e>$ 表示错分类代价的均值。写出在联想概率 $P(\mathbf{x}\in\mathscr{R}_j)$的熵为一常值 H 的约束下最小化 $<P_e>$ 的 Lagrange 函数。 [602]

第12章 神经动态规划

12.1 简介

在第2章，我们认识学习的两种主要范例：有教师学习和无教师学习。无教师学习的范例又可以细分为自组织(无监督)学习和增强式(reinforcement)学习。从第4章到第7章，讨论有教师学习或监督学习的不同形式，从第8章到第11章讨论监督学习的不同形式。在这一章里，我们讨论增强式学习。

监督学习是在"教师"教导下进行的"认知"学习问题：它依赖于一组恰当输入-输出样本的可用性，这些样本能够反映运行环境。与此相反，增强式学习是一种"行为"学习问题：通过学习系统和环境的交互作用完成任务，尽管存在不确定性，但学习系统仍然希望在环境中达到特定目标(Barto et al.,1983;Sutton and Barto,1998)。无教师情况下进行的交互使得增强式学习特别适合代价很高或很难(如果不是不可能)找到一组满意的输入-输出样本的动态情况。

有两种途径研究增强式学习[1]，概述如下：

1. *传统方法*。通过惩罚和奖励的过程进行学习以期达到高度熟练行为的目标。

2. *现代方法*。它基于称为动态规划的一种数学方法，通过考虑将来可能的但实际并未经历的阶段而决定一系列的行动；这里强调的是规划(planning)。

我们讨论的重点是现代增强式学习。

动态规划(dynamic programming)[2]技术处理的是这样一种情况：分阶段做决策，在作下一个决策之前在某种程度上能够预测每个决策的结果。这种情况的一个关键方面是不能孤立地做出决策。相反，现在对低代价的希望必须被将来高代价的失望所抵消(即不能仅追求当前的低代价)。这是一个*信任赋值*(credit assinment)*问题*，因为信任或责任必须赋值给一组相互作用的决策中的每一个决策。为了最优的规划，需要在眼前代价和将来代价中取得有效的折中。这种折中确实被动态规划的形式抓住。特别，动态规划解决一个问题：当可能需要牺牲短期性能的情况下，系统怎样学习提高长期性能？

遵循Bertsekas and Tsitsiklis(1996)，我们称增强式学习的现代方法为*神经动态规划*。这样做主要有两点原因：

- 动态规划提供它的理论基础。
- 神经网络提供它的学习能力。

神经动态规划一个简洁明确的定义是(Bertsekas and Tsitsiklis,1996)：

神经动态规划使一个系统通过观察自身的行为来学会怎样做出好的决策，并且使它能通过使用增强式嵌入机制以改进自己的行动。

在离线方式下使用Monte Carlo仿真可以得到对行为的观察。使用迭代的优化系统通过增强获得对行动的提高。

本章的组织

动态规划有两个主要特征：一个固有的离散时间动态系统，和一个时间上叠加的代价函数。12.2 节讨论这两个特征。随后在 12.3 节讨论 Bellman 最优性方程的公式，它在动态规划中扮演很重要的角色。在 12.4 节和 12.5 节，讨论动态规划计算最佳策略的两种不同方法，它们是策略迭代和值迭代。

在 12.6 节，我们给出神经动态规划涉及的问题的综述。这个综述导致对逼近策略迭代和 Q－学习的讨论，这使它们适宜利用神经网络实现函数逼近。这两个算法将分别在 12.7 节和 12.8 节讨论。12.9 节提出一个使用 Q－学习的计算机实验。

本章在 12.10 节给出最后的评价作为结束。

12.2 Markov 决策过程

考虑一个学习系统或主体(agent)以图 12-1 的方式和环境相互作用。系统依照一个有限的离散时间 Markov 决策过程运行，这个 Markov 决策过程有以下特性：

604

- 环境依概率占据一组有限的离散状态而演化。但是注意状态并不包含过去的统计特性，尽管过去的统计特性对学习系统是有用的。
- 对于每一个环境状态，学习系统可以采取一组有限的可能行动。
- 每当学习系统采取一次行动，就会引起一定的代价。
- 观察状态、采取行动和引发代价都是在离散的时间里发生的。

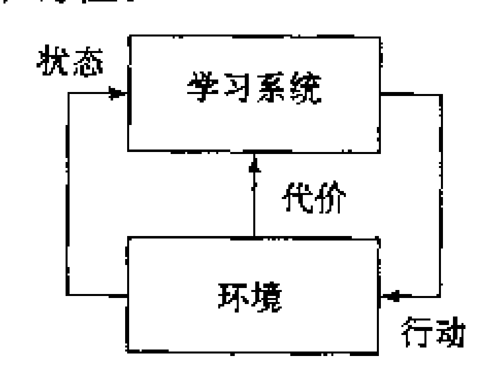


图 12-1 学习系统与环境交互的框图

在当前讨论的背景下，环境的状态定义为学习系统从它和环境交互中获得的过去全部经历的总和，它包含诸如学习系统预测环境未来行为所必需的信息。设表示在时间步 n 的状态的随机变量为 X_n，在时间步 n 的实际状态为 $\mathbf{x}(n)$。有限个状态的集合用$\mathscr{X}$表示。动态规划令人惊奇的一个特点是它的适用性很少依赖状态的性质。因此我们可以不对状态空间结构作任何假设而进行。

例如，对于状态 i，一组可采取的行动(即学习系统作用于环境的输入)设为$\mathscr{A}_i = \{a_{ik}\}$，这里的学习系统采取的行动 a_{ik} 的第二个下标 k 仅仅说明当环境在状态 i 时，可以有不止一个可能的行动。例如，采取行动 a_{ik} 将环境状态从 i 变化到 j 状态本质上为概率性的。然而，最重要的是，从状态 i 到状态 j 的转移概率完全依赖于当前状态 i 和相应的行动 a_{ik}。这就是我们在第 11 章里讨论的 Markov 性质。这个性质是很关键的，因为它意味着环境的当前状态为学习系统提供必需的信息以决定采取什么行动。

用一个随机变量 A_n 表示学习系统在在时间步 n 时采取的行动。用 $p_{ij}(a)$表示在时间步 n 时由于采取行动 a 而使从 i 状态转移到 j 状态的转移概率，其中 $A_n = a$。由 Markov 性质我们有

$$p_{ij}(a) = P(X_{n+1} = j \mid X_n = i, A_n = a) \tag{12.1}$$

由概率论，转移概率 $p_{ij}(a)$必须满足以下两个条件：

$$p_{ij}(a) \geqslant 0 \qquad 对于所有\ i\ 和\ j \tag{12.2}$$

$$\sum_j p_{ij}(a) = 1 \qquad 对于所有\ i \tag{12.3}$$

对于给定数目的状态和转移概率，学习系统随时间采取行动产生的环境状态序列形成一个 Markov 链。我们在第 11 章讨论过 Markov 链。

605

当从一个状态转移到另一个状态时，学习系统招致一个代价。因此在行动 a_{ik} 作用下产生的从状态 i 到状态 j 的第 n 步转移，学习系统招致的代价表示为 $\gamma^n g(i, a_{ik}, j)$，这里的 $g(\cdot,\cdot,\cdot)$ 是一个规定的函数，γ 是*折扣因子*(discount factor)，$0 \leqslant \gamma < 1$。通过调节 γ，可以控制学习系统对它自己行动的短期和长期结果考虑的程度。在极端情况，当 $\gamma = 0$ 系统是短视的，它只考虑它的行动的当前结果。以后将忽略这种极端值，也就是限于讨论 $0 < \gamma < 1$。当 γ 接近 1 时，未来的代价在采取最优行动时变得更为重要。

我们的兴趣在于形成一种*策略*(policy)，这里策略指的是状态到行动的映射。也就是说，给出环境当前状态的知识，一个策略是学习系统决定做什么所使用的一个规则。策略表示为

$$\pi = \{\mu_0, \mu_1, \mu_2, \cdots\} \tag{12.4}$$

其中 μ_n 指的是在时间步 $n = 0, 1, 2, \cdots$，状态 $X_n = i$ 到行动 $A_n = a$ 的映射。这个映射满足

$$\mu_n(i) \in \mathscr{A}_i \qquad 对所有状态\ i \in \mathscr{X}$$

这里 $\mathscr{A}_i$ 表示在状态 i 时学习系统能够采取的行动集合。这样的策略是允许的。

策略可以是不稳定的或稳定的。*不稳定的*(nonstationary)策略是随时间变化的，正如公式(12.4)所示。但当策略不随时间变化时，即

$$\pi = \{\mu, \mu, \mu, \cdots\}$$

就说策略是*稳定的*(stationary)。换句话说，稳定的策略每次遇到一个特定的状态时采取相同的行动。对于稳定的策略，固有的 Markov 链既可以是不平稳的也可以是平稳的。在不平稳的 Markov 链上也可使用稳定的策略，但这是不太明智的。如果使用稳定的策略 μ，那么状态序列 $\{X_n, n = 0, 1, 2, \cdots\}$ 形成一 Markov 链，其转移概率为 $p_{ij}(\mu(i))$，$\mu(i)$ 表示一个行动。由于这个原因这个过程称为 *Markov 决策过程*。

基本问题

动态规划问题分为有限范围和无限范围两种。*有限范围*(finite-horizon)问题中在有限的阶段内对代价累积。*无限范围*(infinite-horizon)问题中在无限的阶段内对代价累积。无限范围问题为有限范围但数目非常大的问题提供一个合理的逼近。因为折扣保证对于任何策略所有状态的代价都是有限的，这样无限范围问题有着特殊的应用。

在无限范围问题中，从初始状态 $X_0 = i$ 开始并使用策略 $\pi = \{\mu_n\}$，总的期望代价定义为

$$J^\pi(i) = E\left[\sum_{n=0}^{\infty} \gamma^n g(X_n, \mu_n(X_n), X_{n+1}) \mid X_0 = i\right] \tag{12.5}$$

606

其中期望值是对 Markov 链 $\{X_1, X_2, \cdots\}$ 取值。函数 $J^\pi(i)$ 叫做策略 π 从状态 i 开始的 cost-to-go 函数。它的最优值记为 $J^*(i)$，定义为

$$J^*(i) = \min_\pi J^\pi(i) \tag{12.6}$$

当策略 π 稳定时，即 $\pi = \{\mu, \mu, \mu, \cdots\}$，我们用符号 $J^\mu(i)$ 代替 $J^\pi(i)$，并当下列条件成立

时说 μ 是最佳的：

$$J^{\mu}(i) = J^{*}(i) \qquad \text{对于所有的初始状态 } i \tag{12.7}$$

我们可以对动态规划的基本问题做如下总结：

给定描述学习系统和环境相互作用的稳定 Markov 决策过程，找到一个稳定的策略 $\pi = \{\mu, \mu, \mu, \cdots\}$ 使对所有的初始状态 i 有最小的 cost-to-go 函数 $J^{\mu}(i)$。

注意，在学习过程中学习系统的行为可以随时间改变。但是学习系统寻找的最优策略是稳定的(Watkins, 1989)。

12.3 Bellman 最优准则

动态规划技术依赖归功于 Bellman(1957)的通称为最优原则(principle of optimality)的非常简单的思想。这个原则可简单陈述为(Bellman and Dreyfus, 1962)：

一个最优策略有这样的性质，无论初始状态和初始决策是什么，对于第一个决策所导致的状态，剩余决策必须成为最优策略。

正如这里使用的那样，一个"决策"(decision)是在特定时间的一种控制选择，一个"策略"(policy)是整个控制序列或控制函数。

为用数学公式表示最优原则，考虑一 有限范围问题，它的 cost-to-go 函数定义为

$$J_0(X_0) = E\left[g_K(X_K) + \sum_{n=0}^{K-1} g_n(X_n, \mu_n(X_n), X_{n+1})\right] \tag{12.8}$$

其中 K 是范围(即阶段数目)，$g_K(X_K)$是最终代价。给定 X_0，式(12.8)中的期望值是对剩余状态 X_1，…，X_{K-1}求出的。现在我们可以正式陈述最优原则如下(Bertsekas, 1995b)：

令 $\pi^* = \{\mu_0^*, \mu_1^*, \cdots, \mu_{K-1}^*\}$ 作为基本有限范围问题的最优策略。假设使用最优策略 π^* 时，给定的状态 X_n 发生的概率为正。考虑当环境在时刻 n 时状态为 X_n 的子问题，假设我们希望最小化对应的 cost-to-go 函数

607

$$J_n(X_n) = E\left[g_K(X_K) + \sum_{k=n}^{K-1} g_k(X_k, \mu_k(X_k), X_{k+1})\right] \tag{12.9}$$

其中 $n = 0, 1, \cdots, K-1$。这时截断策略 $\{\mu_n^*, \mu_{n+1}^*, \cdots, \mu_{K-1}^*\}$ 对于子问题是最优的。

通过下面的讨论，我们可以直观地说明最优原则的合理性：如果截断策略 $\{\mu_n^*, \mu_{n+1}^*, \cdots, \mu_{K-1}^*\}$ 不是如陈述的那样为最优，那么一旦在 n 时刻到达 X_n 状态，通过简单转换到对于子问题最优的策略，我们可以减少 cost-to-go 函数 $J_n(X_n)$。

最优原则基于"分而治之"(divide and conquer)的工程概念。基本上，一个复杂的多阶段规划或控制问题的最优策略，可通过以下处理构造：

- 构造一个仅包含系统最后一个阶段的"尾部子问题"(tail subproblem)的最优策略。
- 扩展最优策略至包含系统最后两个阶段的"尾部子问题"。
- 以这种方式继续这种过程，直到处理完整个问题。

动态规划算法

在前面描述过程的基础上，我们可以提出动态规划算法，它从时期 $N-1$ 到时期 0 反向

处理。令 $\pi=\{\mu_0,\mu_1,\cdots,\mu_{K-1}\}$表示允许策略。对每一个 $n=0, 1, \cdots, K-1$，令 $\pi^n=\{\mu_n,\mu_{n+1},\cdots\mu_{K-1}\}$，令 $J_n^*(X_n)$表示从时间 n 的状态 X_n 开始到时间 K 的$(K-n)$阶段问题的最优代价；即

$$J_n^*(X_n) = \min_{\pi^n} \mathop{E}_{(X_{n+1},\cdots,X_{K-1})}\left[g_K(X_K)+\sum_{k=n}^{K-1}g_k(X_k,\mu_k(X_k),X_{k+1})\right] \tag{12.10}$$

它表示式(12.9)的最优形式。考虑到 $\pi^n=(\mu_n,\pi^{n+1})$和部分展开式(12.10)的右边和，我们可以写成

$$\begin{aligned}J_n^*(X_n) &= \min_{(\mu_n,\pi^{n+1})} \mathop{E}_{(X_{n+1},\cdots,X_{K-1})}\Big[g_n(X_n,\mu_n(X_n),X_{n+1}) \\ &\qquad + g_K(X_K)+\sum_{k=n+1}^{K-1}g_k(X_k,\mu_k(X_k),X_{k+1})\Big] \\ &= \min_{\mu_n}\mathop{E}_{X_{n+1}}\Big\{g_n(X_n),\mu_n(X_n),X_{n+1}) \\ &\qquad + \min_{\pi^n}\mathop{E}_{(X_{n+2},\cdots,X_{K-1})}\Big[g_K(X_K)+\sum_{k=n+1}^{K-1}g_k(X_k,\mu_k(X_k),X_{k+1})\Big]\Big\} \\ &= \min_{\mu_n}\mathop{E}_{X_{n+1}}\Big[g_n(X_n,\mu_n(X_n),X_{n+1})+J_{n+1}^*(X_{n+1})\Big]\end{aligned} \tag{12.11}$$

在最后一行，使用了式(12.10)的定义，以 $n+1$ 代替 n。现在我们假设对某一 n 和所有的 X_{n+1}有

$$J_{n+1}^*(X_{n+1}) = J_{n+1}(X_{n+1}) \tag{12.12}$$

608

那么式(12.11)可写成

$$J_n^*(X_n) = \min_{\mu_n}\mathop{E}_{X_{n+1}}[g_n(X_n,\mu_n(X_n),X_{n+1})+J_{n+1}(X_{n+1})] \tag{12.13}$$

如果式(12.12)对所有 X_{n+1}成立，那么下式对所有 X_n 也显然成立：

$$J_n^*(X_n) = J_n(X_n)$$

因此，可从(12.13)导出

$$J_n(X_n) = \min_{\mu_n}\mathop{E}_{X_{n+1}}[g_n(X_n,\mu_n(X_n),X_{n+1})+J_{n+1}(X_{n+1})]$$

我们可以正式陈述动态规划算法如下(Bertsekas, 1995b)：

对每一个初始状态 X_0，基本有限范围问题的最优代价 $J^*(X_0)$等于 $J_0(X_0)$，其中函数 J_0 从下面算法的最后一步得到：

$$J_n(X_n) = \min_{\mu_n}\mathop{E}_{X_{n+1}}[g_n(X_n,\mu_n(X_n),X_{n+1})+J_{n+1}(X_{n+1})] \tag{12.14}$$

按时间反向运行，且

$$J_K(X_K) = g_K(X_K)$$

另外，若 μ_n^* 使得式(12.14)的右边对于任意 n 和 X_n 为最小，那么策略 $\pi^*=\{\mu_0^*,\mu_1^*,\cdots,\mu_{K-1}^*\}$是最优的。

Bellman 最优性方程

以其基本形式，动态规划算法处理有限范围问题。我们感兴趣的是推广这个算法的用

途，即处理在稳定策略 $\pi=\{\mu,\mu,\mu,\cdots\}$情况下，式(12.5)的 cost-to-go 函数所描述的无限范围折扣问题。为了达到这一点，我们做下面两件事：

- 反转算法的时间索引，使得它和折扣问题对应。
- 定义代价 $g_n(X_n,\mu(X_n),X_{n+1})$如下：

$$g_n(X_n,\mu(X_n),X_{n+1}) = \gamma^n g(X_n,\mu(X_n),X_{n+1}) \tag{12.15}$$

现在可以重新定义动态规划算法如下(参看习题 12.4)：

$$J_{n+1}(X_0) = \min_{\mu} \underset{X_1}{E}[g(X_0,\mu(X_0),X_1) + \gamma J_n(X_1)] \tag{12.16}$$

它从初始条件

$$J_0(X) = 0 \qquad 对所有 X$$

开始状态 X_0 是初始状态，X_1 是策略 μ 的行动导致的新状态，γ 是折扣因子。

令 $J^*(i)$表示对初始状态 $X_0=i$ 的最优无限范围的代价。我们可以把 $J^*(i)$看作相应的 K 阶段最优代价 $J_K(i)$当 K 趋于无穷大时的极限；即

609

$$J^*(i) = \lim_{K\to\infty} J_K(i) \qquad 对所有 i \tag{12.17}$$

这个关系联系着有限范围和无限范围之间的折扣问题。在式(12.16)中，置 $n+1=K$，$X_0=i$，并应用式(12.17)，我们得到

$$J^*(i) = \min_{\mu} \underset{X_1}{E}[g(i,\mu(i),X_1) + \gamma J^*(X_1)] \tag{12.18}$$

为了计算最优无限范围代价 $J^*(i)$的值，按下面两个阶段进行处理：

1. 计算代价 $g(i,\mu(i),X_1)$对 X_1 的期望值，

$$E[g(i),\mu(i),X_1] = \sum_{j=1}^{N} p_{ij} g(i,\mu(i),j) \tag{12.19}$$

其中 N 是环境状态的数目，p_{ij}是初始状态 $X_0=i$ 到新状态 $X_1=j$ 的转移概率。式(12.19)定义的量是在状态 $X_0=i$ 使用策略 μ 建议的行动引起的*立即期望代价*。利用 $c(i,\mu(i))$表示这个代价，可以写为

$$c(i,\mu(i)) = \sum_{j=1}^{N} p_{ij} g(i,\mu(i),j) \tag{12.20}$$

2. 计算 $J^*(X_1)$对 X_1 的期望值。这里注意，如果知道有限状态系统的每一个状态 X_1 的代价 $J^*(X_1)$，我们可以根据固有的 Markov 链的转移概率决定 $J^*(X_1)$的期望值如下：

$$E[J^*(X_1)] = \sum_{j=1}^{N} p_{ij} J^*(j) \tag{12.21}$$

这样，将式(12.19)至(12.21)代入式(12.16)，我们得到期望的结果

$$J^*(i) = \min_{\mu}\left(c(i,\mu(i)) + \gamma \sum_{j=1}^{N} p_{ij}(\mu) J^*(j)\right) \qquad i = 1,2,\cdots,N \tag{12.22}$$

式(12.22)叫做 Bellman *最优性方程*。它不应该被看作算法。相反，它表示 N 个方程组，每个方程对应一个状态。这个方程组的解定义环境 N 个状态的最优 cost-to-go 函数。

有两种计算最优策略基本方法。它们称为策略迭代和值迭代。这两种方法分别在 12.4 节和 12.5 节讨论。

12.4 策略迭代

我们开始描述策略迭代算法，首先介绍 Watkins(1989)提出的 Q－因子的概念。考虑一个现有的策略 μ，它的所有状态 i 的 cost-to-go 函数 $J^{\mu}(i)$为已知。对每一个状态 $i\in\mathscr{X}$和行动 $a\in\mathscr{A}_i$，Q－因子定义为立即代价加上遵循策略 μ 的所有后继状态的折扣代价之和，表示为 [610]

$$Q^{\mu}(i,a)=c(i,a)+\gamma\sum_{j=1}^{n}p_{ij}(a)J^{\mu}(j) \tag{12.23}$$

其中行动 $a=\mu(i)$。注意 Q－因子 $Q^{\mu}(i,a)$比 cost-to-go 函数 $J^{\mu}(i)$包含的信息更多。例如，行动可以只依靠 Q－因子来排序，而依靠 cost-to-go 函数排序时还需状态转移概率和代价的知识。

通过设想由初始状态 1，2，…，N 和所有状态－行动对(i，a)组成其状态的新系统，如图 12-2 所描绘，我们可以深入了解 Q－因子的含义。有两种可能发生的不同概率：

- 系统在状态(i,a)，在这种状况下，不采取行动。以概率 $p_{ij}(a)$自动转变为状态 j；同时招致代价 $g(i,a,j)$。
- 系统在状态 i，在这种状况下，采取行动 $a\in\mathscr{A}_i$ 后。下一个确定性状态是(i,a)。

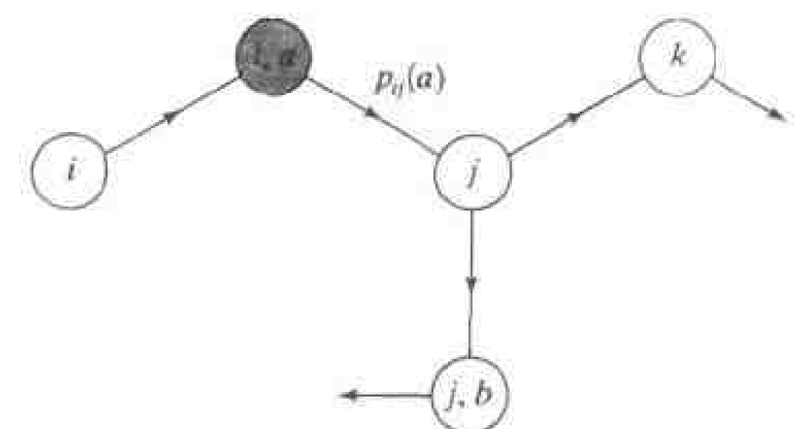


图 12-2 两个可能的转移：从状态(i,a)到状态 j 的转移为概率性的，但从状态 i 到状态(i,a)的转移为确定性的

我们说策略 μ 对 cost-to-go 函数 $J^{\mu}(i)$是贪心的，如果对所有的状态，$\mu(i)$是满足下列条件的行动：

$$Q^{\mu}(i,\mu(i))=\min_{a\in\mathscr{A}_i}Q^{\mu}(i,a) \qquad \text{对所有 } i \tag{12.24}$$

对式(12.24)的下列两点观察得注意：

- 可能有多于一个行动最小化某一状态的 Q－因子集合，在这种情况下，对于有关的 cost-to-go 函数可以有多于一个的贪心策略。
- 不同的 cost-to-go 函数可能有一个相同的贪心策略。

另外，下面的事实对所有动态规划方法都是基本的：

$$Q^{\mu^*}(i,\mu^*(i))=\min_{a\in\mathscr{A}_i}Q^{\mu^*}(i,a) \tag{12.25}$$

这里 μ^*是最优策略，J^*是相应的最优 cost-to-go 函数。

用我们处理 Q－因子和贪心策略的概念，可以描述**策略迭代**(policy iteration)**算法**。特别地，算法交替在下面两个步骤中运行(Bertsekas，1995b)：

1. **策略求值步骤**，在这个步骤里，对所有状态和行动求当前策略的 cost-to-go 函数值和相应的 Q－因子的值。

2. **策略改进步骤**，更新当前策略使其成为第一步计算出的 cost-to-go 函数的贪心策略。

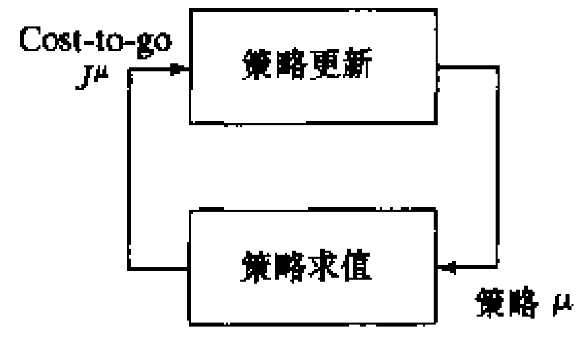


图 12-3 策略迭代算法框图

这两个步骤见图 12-3。具体地，我们从某一初始策略 μ_0 开始，然后产生一系列新策略 μ_1，μ_2，…。设当前策略 [611]

为 μ_n，我们执行策略求值步骤时，计算 cost-to-go 函数 $J^{\mu_n}(i)$，作为下列线性方程组的解(参看式(12.22))：

$$J^{\mu_n}(i) = c(i,\mu_n(i)) + \gamma \sum_{j=1}^{N} p_{ij}(\mu_n(i)) J^{\mu_n}(j) \qquad i = 1,2,\cdots,N \tag{12.26}$$

其中 $J^{\mu_n}(1)$，$J^{\mu_n}(2)$，…，$J^{\mu_n}(N)$是未知数. 使用这些结果，我们对状态-行动对(i, a)计算 Q-因子(参看式(12.23))

$$Q^{\mu_n}(i,a) = c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^{\mu_n}(j) \quad a \in \mathscr{A}_i \qquad i = 1,2,\cdots,N \tag{12.27}$$

接着，通过计算如下定义的新策略 μ_{n+1}来完成策略改进：

$$\mu_{n+1}(i) = \arg\min_{a \in \mathscr{A}_i} Q^{\mu_n}(i,a) \qquad i = 1,2,\cdots,N \tag{12.28}$$

利用策略 μ_{n+1}代替 μ_n，重复刚才描述的两个步骤直到有

$$J^{\mu_{n+1}}(i) = J^{\mu_n}(i) \qquad 对所有\ i$$

此时终止算法于策略 μ_n。由于 $J^{\mu_{n+1}} \leqslant J^{\mu_n}$(参看习题12.5)，可以说经过有限次迭代后策略迭代算法会结束，因为固有的 Markov 决策过程仅有有限数目的状态。表 12-1 概括基于式(12.26)和(12.28)的策略迭代算法。

表 12-1 策略迭代算法小结

1. 从任意的初始策略 μ_0 开始。
2. 对所有的状态 $i \in \mathscr{X}$和行动 $a \in \mathscr{A}_i$，计算 $J^{\mu_n}(i)$和 $Q^{\mu_n}(i, a)$，$n=0, 1, 2, \cdots$。
3. 对每一个状态 i，计算
$$\mu_{n+1}(i) = \arg\min_{a \in \mathscr{A}_i} Q^{\mu_n}(i,a)$$
4. 重复第2、3步，直到 μ_{n+1}与 μ_n 无差别，那时的 μ_n 就是所求的策略。

12.5 值迭代

在策略迭代算法中，算法每次迭代过程必须重新计算整个 cost-to-go 函数，这样代价是很高的。即使新策略和旧策略的 cost-to-go 函数很相似，很遗憾这个计算也没有显著的改进。然而，有另外一种用于寻找最优策略的方法能够在计算 cost-to-go 函数时避免烦琐的重复计算。这个以逐次逼近为基础的替代方法就是值迭代算法。

612

值迭代(value iteration)算法涉及对一序列有限范围问题中的每一个求解 Bellman 最优性方程(12.22)。当算法的迭代数目趋于无穷时，在极限处有限范围问题的 cost-to-go 函数对所有的状态一致收敛于相应的无限范围问题的 cost-to-go 函数(Ross, 1983; Bertsekas, 1995b)。

令 $J_n(i)$表示在值迭代算法中迭代 n 时对状态 i 的 cost-to-go 函数。算法从任意的猜测$J_0(i)$开始，$i=1, 2, \cdots, N$。$J_0(i)$的惟一约束是它应该有界；对于有限范围问题，这是自动成立的。如果最优 cost-to-go 函数 $J^*(i)$的某一估计可用，那么它应该被用作初始值$J_0(i)$。一旦选择了 $J_0(i)$，就可以计算 cost-to-go 函数序列 $J_1(i)$，$J_2(i)$，…，使用值迭代算法

$$J_{n+1}(i) = \min_{a \in \mathscr{A}_i}\left\{c(i,a) + \gamma\sum_{j=1}^{N} p_{ij}(a) J_n(j)\right\}, \qquad i = 1,2,\cdots,N \tag{12.29}$$

对于状态 i 应用式(12.29)描述的 cost-to-go 函数的更新，这称为 i 的代价的支持(backing up of i's cost)。这个支持是 Bellman 最优性方程(12.22)的直接实现。注意对状态 $i=1, 2, \cdots, N$，式(12.29)中 cost-to-go 函数的值在算法的每一次迭代时同时更新。这个实现方法表示值迭代算法传统的同步形式[3]。这样，从任意的初始值 $J_0(1)$，$J_0(2)$，…，$J_0(N)$开始，当迭代数目 n 趋近无穷时，式(12.29)描述的算法将收敛于相应的最优值 $J^*(1)$，$J^*(2)$，…，$J^*(N)$(Ross, 1983; Bertsekas, 1995b)。

与策略迭代算法不同的是，在值迭代算法中不是直接计算最优策略，而是首先用式(12.29)计算最优值 $J^*(1)$，$J^*(2)$，…，$J^*(N)$，然后获得关于该最优集合的贪心策略作为最优策略。就是说，

$$\mu^*(i) = \arg\min_{a \in \mathscr{A}_i} Q^*(i,a), \qquad i = 1,2,\cdots,N \tag{12.30}$$

613

这里
$$Q^*(i, a) = c(i, a) + \gamma\sum_{j=1}^{N} p_{ij}(a) J^*(j), \qquad i = 1, 2, \cdots, N \tag{12.31}$$

表 12-2 给出基于式(12.29)至(12.31)的值迭代算法的小结，其中包括式(12.29)的停止准则。

表 12-2 值迭代算法小结

1. 从状态 $i=1, 2, \cdots, N$ 的任意初始值 $J_0(i)$ 开始。
2. 对 $n=0, 1, 2, \cdots$，计算

$$J_{n+1}(i) = \min_{a \in \mathscr{A}_i}\left\{c(i,a) + \gamma\sum_{j=1}^{N} p_{ij}(a) J_n(j),\right\}, \quad a \in \mathscr{A}_i, \; i = 1,2,\cdots,N$$

重复这种操作直到

$$|J_{n+1}(i) - J_n(i)| < \epsilon \qquad \text{对每一个状态 } i$$

这里的 ϵ 是指定的容许参数。假定 ϵ 足够小，使 $J_n(i)$ 充分接近最优 cost-to-go 函数 $J^*(i)$。因此我们可以置

$$J_n(i) = J^*(i) \qquad \text{对所有状态 } i$$

3. 计算 Q－因子

$$Q^*(i,a) = c(i,a) + \gamma\sum_{j=1}^{N} p_{ij}(a) J^*(j) \qquad \text{对 } a \in \mathscr{A}_i \text{ 且 } i = 1,2,\cdots,N$$

由此，确定贪心策略作为 $J^*(i)$ 的最优策略：

$$\mu^*(i) = \arg\min_{a \in \mathscr{A}_i} Q^*(i,a)$$

例 12.1 驿车问题 为了说明 Q－因子在动态规划中的作用，我们考虑驿车(stagecoach problem)问题。在 19 世纪中叶密苏里的一个幸运追求者决定去西部加入在加里福尼亚的淘金潮(Hiller and Lieberman, 1995)。行程需要乘驿车穿过不安全的乡村，沿途会有强盗攻击的危险。行程的起始点(密苏里州)和终点(加里福尼亚州)是固定的。但是有很多可以选择的路径，有可能经过其他 8 个州，如图 12-4 所示。在图中，我们有以下规定：

- 一共 10 个州，每个州用一个字母表示。
- 行进的方向是从左到右。
- 从开始的州 A(密苏里州) 到终点的州 J(加里福尼亚州) 有 4 个阶段。
- 幸运追求者从一个州到下一个州行动是向上(Up)、向下(Down)或直接向前(Straight)。

614

- 从 A 到 J 一共有 18 条可能路径。

图 12-4 还包括对每一条路径的人身保险策略的代价，选择每一条路线是基于对该路线的安全代价的仔细评估。问题是从 A 到 J 找到一条人身保险最便宜的路线。

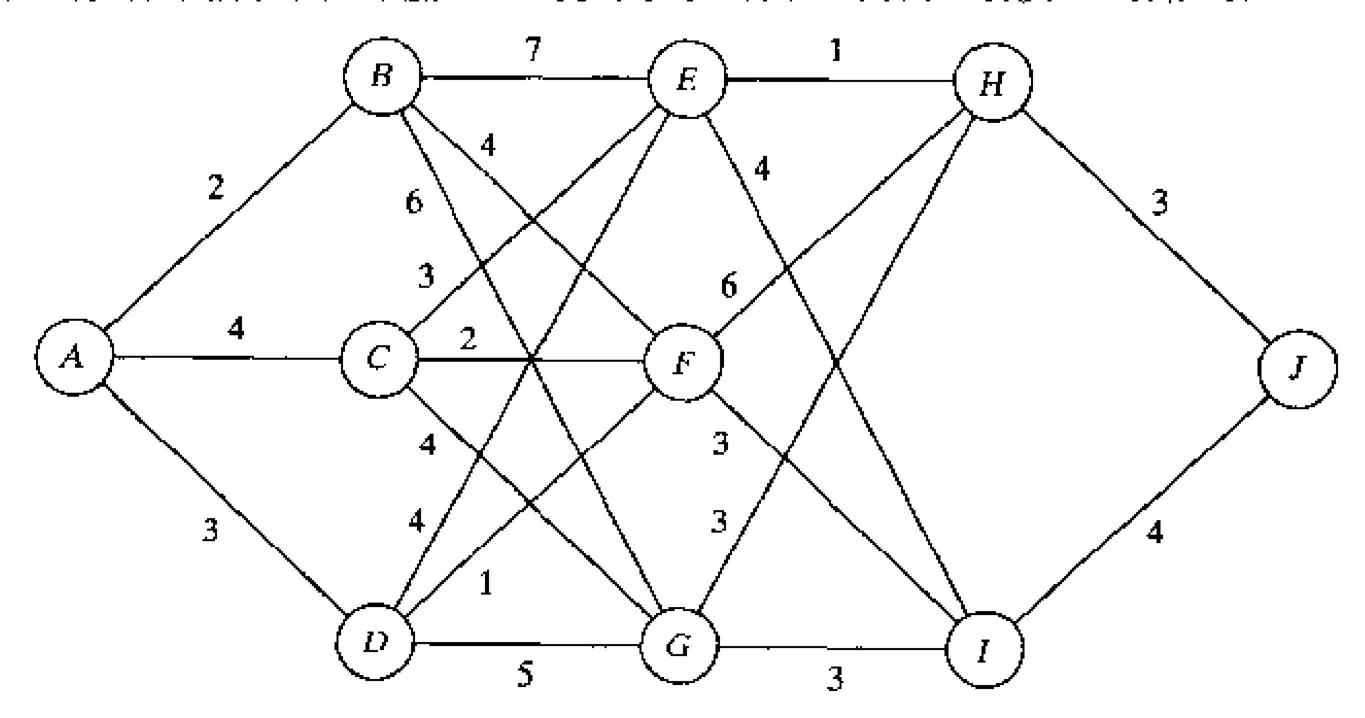


图 12-4 驿车问题的流向图

为了找到最优路线，我们从终点 J 开始向后推演，考虑一系列有限范围问题。这符合 12.3 节的 Bellman 最优性原则。

计算终点前的最后一阶段的 Q - 因子，从图 12-5a 可以得出终点 Q - 值如下：

$Q(H,\text{down})$	=	3
$Q(I,\text{up})$	=	4

这些数值从图 12-5a 可以分别得出。

然后向后再移动一阶段，使用图 12-5a 得出的 Q - 值，计算下面的 Q - 值：

$Q(E,\text{straight})$	=	$1+3=4$
$Q(E,\text{down})$	=	$4+4=8$
$Q(F,\text{up})$	=	$6+3=9$
$Q(F,\text{down})$	=	$3+4=7$
$Q(G,\text{up})$	=	$3+3=6$
$Q(G,\text{straight})$	=	$3+4=7$

由于需要找到最小保险策略的路径，Q - 值表明只有 $E\rightarrow H$、$F\rightarrow I$ 和 $G\rightarrow H$ 路径应保留，而其他路径应删除，如图 12-5b。

再向后移动一阶段，对状态 B，C，D 重复这种 Q - 因子计算，保留那些有最低安全评价的路径，就得到图 12-5c。

615

最后，向后移动到第一阶段，重复上面的计算，就得到图 12-5d。从图中我们看到共有 3 条最优路径如下：

$$A\rightarrow C\rightarrow E\rightarrow H\rightarrow J$$
$$A\rightarrow D\rightarrow E\rightarrow H\rightarrow J$$
$$A\rightarrow D\rightarrow F\rightarrow I\rightarrow J$$

它们产生的总体代价都是 11。

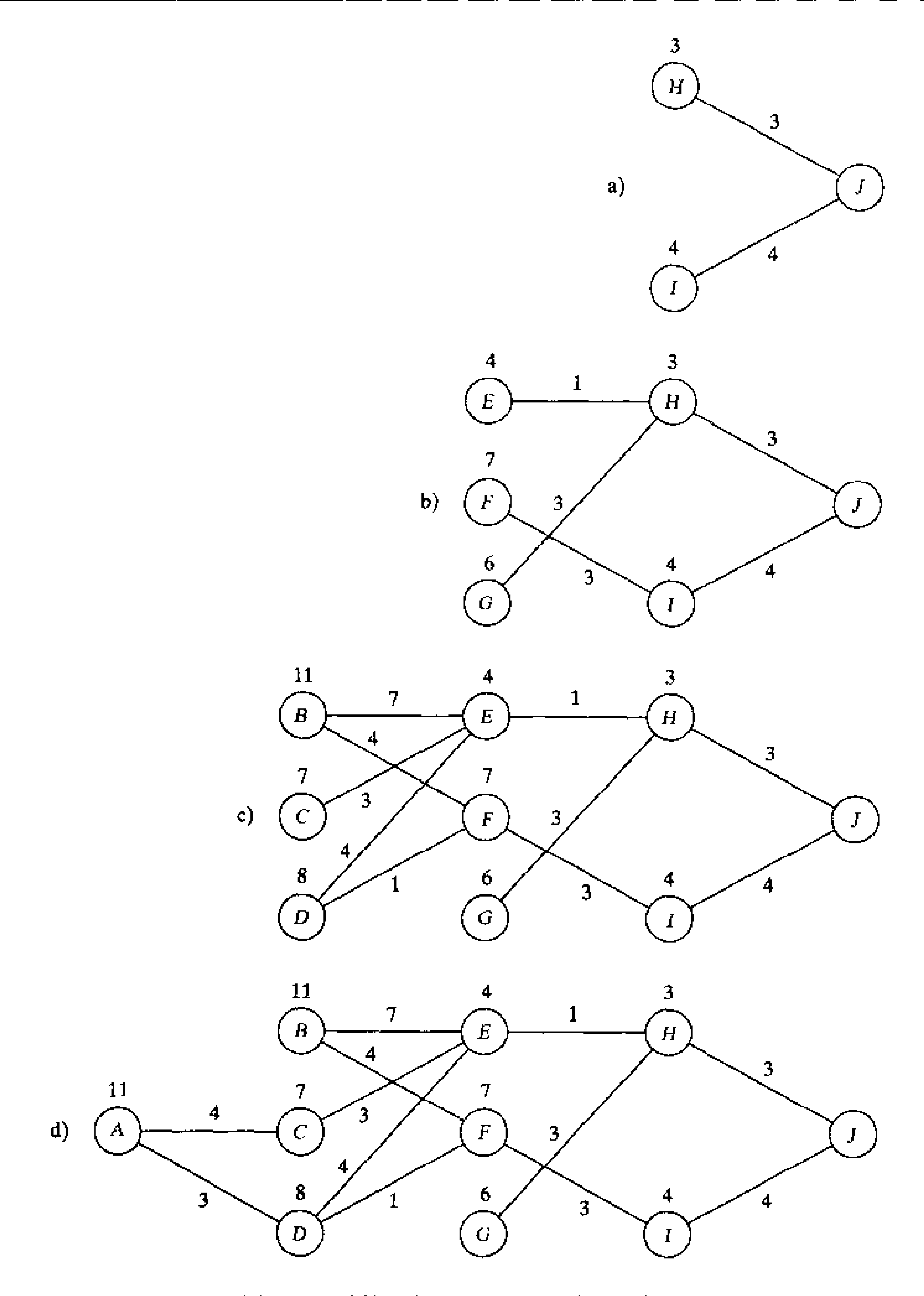


图 12-5 计算驿车问题 Q-因子涉及的步骤

616

12.6 神经动态规划

动态规划的主要目标是寻找一种最优策略，即学习系统对环境每个可能状态应该采取行动的最优选择。在这种环境中，当考虑利用策略迭代或值迭代算法求解一个动态规划问题时，必须记住两个实际问题：

- 维数灾。对现实世界的许多困难问题，可能的状态和允许的行动数目如此之大，以

致动态规划所需计算量是不堪忍受的。对于涉及总共 N 个可能状态和对每个状态有 M 个允许行动的动态规划问题，例如，一个稳定策略的值迭代算法的每步迭代需要 N^2M 次。当 N 很大时，这常常使得即使是完成算法的一次迭代也不可能。例如，十五子棋(backgammon)有 10^{20} 个状态，这意味着算法的一次迭代利用 1000 MIPS 处理器也需 1000 年(Barto, et al, 1995)。

- 不完全信息。策略迭代或值迭代算法要求有 Markov 决策过程的固有先验知识。即为了最优策略的计算可行，我们需要知道状态转移概率 p_{ij} 和观察代价 $g(i,a,j)$。遗憾的是，这些先验知识并非总是可得的。

由于这两个困难中的任何一个或全部，我们常常放弃最优策略而使用次优策略。

这里我们感兴趣的是在次优过程中为逼近最优 cost-to-go 函数 $J^*(i)$，$i\in\mathscr{X}$ 这个目的而涉及神经网络的使用或(和)模拟。特别地，对一特定状态 i，$J^*(i)$ 由它的合适逼近 $\tilde{J}(i,\mathbf{w})$ 所代替，其中 $\mathbf{w}$ 是参数向量。函数 $\tilde{J}(\cdot,\mathbf{w})$ 称为*评分函数*(scoring function)或*近似 cost-to-go 函数*，函数的值 $\tilde{J}(i,\mathbf{w})$ 称为状态 i 的*分数*(scores)或*近似 cost-to-go 代价*。因此在图 12-6 中，分数 $\tilde{J}(i,\mathbf{w})$ 为输入状态 i 时神经网络的输出。这里利用的是所谓*通用逼近*，正如在前面几章中所讨论的那样，它是多层感知器和径向基函数网络的固有特征。

我们有特别兴趣的动态规划问题是那些具有大量状态而要求寻找有较小维数的参数向量 $\mathbf{w}$ 的评分函数 $\tilde{J}(\cdot,\mathbf{w})$。这种形式逼近称为*紧凑表示*，仅需存储参数向量 $\mathbf{w}$ 和评分函数 $\tilde{J}(\cdot,\mathbf{w})$ 的一般结构。对所有状态 $i\in\mathscr{X}$ 只有需要时才产生分数 $\tilde{J}(i,\mathbf{w})$。对于给定的神经网络结构(例如多层感知器)，问题是寻找参数向量 $\mathbf{w}$，使得对所有 $i\in\mathscr{X}$ 分数 $\tilde{J}(i,\mathbf{w})$ 提供最优值 $J^*(i)$ 的一个满意的逼近。

617

由第 4 章至第 7 章给出的关于有教师学习的材料，我们知道，不管一个神经网络的类型如何，都要求有一个表示该任务的标定数据集。但是，在动态规划问题的背景下，没有这样的训练数据(即输入－输出样本$\{(i,\tilde{J}^*(i))\}$可用来训练图 12-6 中的神经网络，使得在某种统计意义下优化它的设计。这样惟一的可能性是利用 Monte Carlo 模拟，这里利用一个替代模型替代基本 Markov 决策过程的实际系统。这样导致一种新的离线动态规划运行方式，它有如下潜在的好处(Bertsekas and Tsitsiklis, 1996)：

1. 利用模拟近似地求最优 cost-to-go 函数的值是区别神经动态规划方法和传统动态规划逼近方法的关键思想。

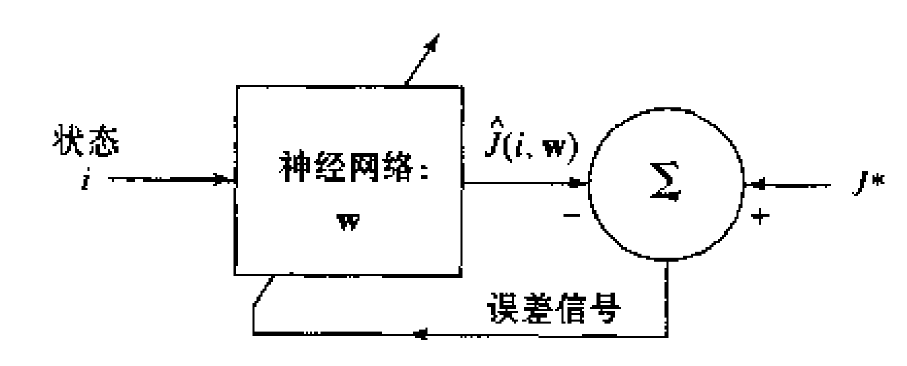


图 12-6 逼近最优 cost-to-go 函数 J^* 的神经网络

2. 模拟允许利用神经动态规划方法设计没有明显模型可用的系统。对于这种系统，传统的动态规划技术是不可能用的，因为提供状态转移概率的估计如果不是不可能那也是很烦琐的。

3. 通过模拟，可以隐式地确认系统中最重要或有代表性的状态，即那些在模拟中被经常访问到的状态。结果，由神经网络发现的评分函数可以对这些特殊状态的最优 cost-to-go 函数提供一个好的逼近。对一个困难的动态规划问题最终结果可能是一个好的次优策略。

但是，重要的是要认识到一旦引入逼近，就不能期望评分函数 $\tilde{J}(\cdot,\mathbf{w})$ 收敛到最优的 cost-to-go 函数 $J^*(\cdot)$。原因很简单，$J^*(\cdot)$ 可能不在选定的神经网络结构所能精确表达的函

数集内

在下面两节里我们讨论两个 cost-to-go 函数逼近的动态规划逼近过程。在 12.7 节描述的第一个过程处理逼近策略迭代，这里假设系统具有可用的 Markov 模型。在 12.8 节描述的第二过程处理一个称为 Q－学习的过程，它仅作一些弱的假设。

618

12.7 逼近策略迭代

假设我们有一动态规划问题，它的可能状态数目和允许的行动数目非常大，使得利用传统处理方法是不现实的。假如我们有该系统的模型，即转移概率 $p_{ij}(a)$ 和观察代价 $g(i,a,j)$ 都是已知的。为了处理这种情况，我们基于下面所述的 Monte Carlo 模拟和最小二乘法提出使用策略迭代的近似(Bertsekas and Tsitsiklis, 1996)。

图 12-7 给出逼近策略迭代算法的简化框图。它相似于图 12-3 所示的传统策略迭代算法框图，但有一个重要的区别：在图 12-3 中的策略求值步骤由它的一个逼近所替代。因此逼近策略迭代算法交替进行如下的逼近策略求值步骤和策略改进步骤：

1. **逼近策略求值步骤**。给定当前策略 μ，对所有状态 i 的实际 cost-to-go 函数 $J^{\mu}(i)$ 计算它的逼近，即 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$。向量 $\mathbf{w}$ 是完成逼近的神经网络参数。

2. **策略改进步骤**。利用逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$ 产生改进的策略 μ。对所有 i，新策略设计对 $\tilde{J}^{\mu}(i,\mathbf{w})$ 是贪心的。

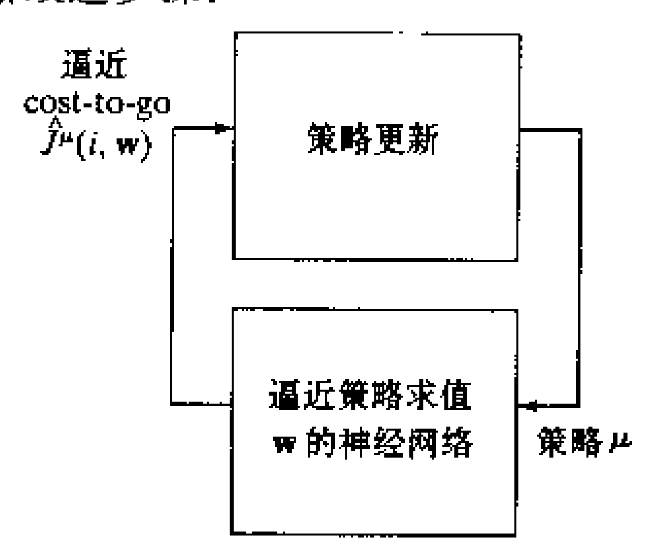


图 12-7 逼近策略迭代算法的简化框图

由于逼近策略迭代算法产生满意解，因此仔细挑选策略初始化算法非常重要。这可利用启发式思想完成。或者我们可以从某个权值向量 $\mathbf{w}$ 开始，用它导出一个贪心策略，接着利用该策略为初始策略。

619

假设除知道转移概率和观察代价之外，我们有如下几项：

- 一个稳定的策略 μ 作为初始策略。
- 一个状态集 $\mathscr{X}$ 代表运行环境。
- 对于每个 $i\in\mathscr{X}$，cost-to-go 函数 $J^{\mu}(i)$ 的 $M(i)$ 个样本集；一个这样的样本记为 $k(i,m)$，其中 $m=1,2,\cdots,M(i)$。

令 $\tilde{J}^{\mu}(i,\mathbf{w})$ 记 cost-to-go 函数 $J^{\mu}(i)$ 的逼近表示。逼近由神经网络完成(例如用反向传播算法训练的多层感知器)。神经网络的参数向量 $\mathbf{w}$ 利用最小二乘法决定，即最小化代价函数

$$\mathscr{E}(\mathbf{w})=\sum_{i\in\mathscr{X}}\sum_{m=1}^{M(i)}(k(i,m)-\tilde{J}^{\mu}(i,\mathbf{w}))^2 \tag{12.32}$$

在确定最优权值向量 $\mathbf{w}$ 从而有逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$ 之后，我们再利用下列公式确定逼近 Q－因子(参看式(12.20)和式(12.23))：

$$Q(i,a,\mathbf{w})=\sum_{j\in\mathscr{X}}p_{ij}(a)(g(i,a,j)+\gamma\tilde{J}^{\mu}(j,\mathbf{w})) \tag{12.33}$$

其中 $p_{ij}(a)$ 为在行动 a(已知)下从状态 i 到状态 j 的转移概率，$g(i,a,j)$ 是观察代价(也为已知)，而 γ 是规定的折扣因子。根据下列公式，通过使用这些逼近 Q－因子确定一种改进策

略以完成迭代(参看(11.26))：

$$\mu(i)=\arg\min_{a\in\mathcal{A}_i} Q(i,a,\mathbf{w}) \tag{12.34}$$

重要的是注意，式(12.33)和(12.34)仅被模拟器用于在由模拟实际访问的状态而不是在所有状态产生行动。正因为如此，这两个公式没有受到维数灾的影响。

图12-8给出一个逼近策略迭代算法的更加详细的框图。这个框图由四个互连的模块组成(Bertsekas and Tsitsiklis,1996)：

1. 模拟器，它利用给定的对状态转移概率和观察到的一步代价构建环境的一个替代模型。模拟器产生两类东西：(a)模拟环境的行动进行响应的状态，(b)对给定策略的cost-to-go函数抽样。

620

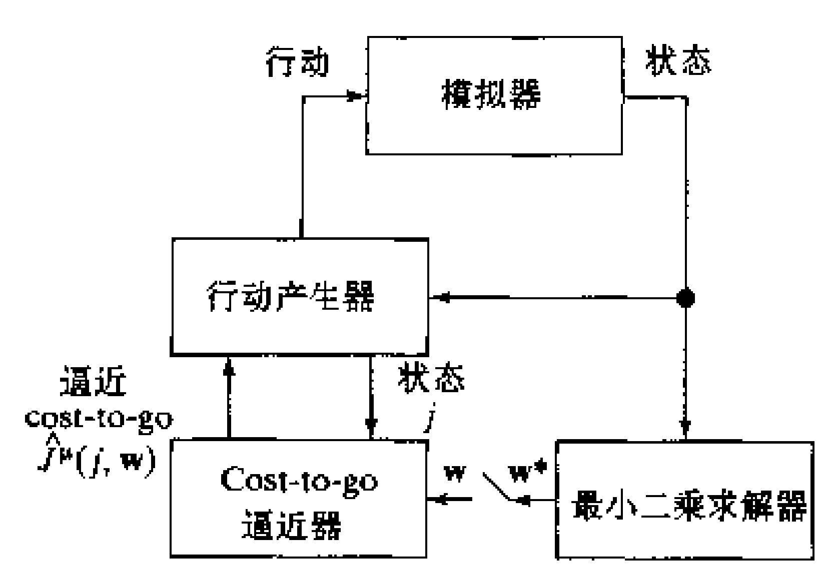


图12-8 逼近策略迭代算法详细设计

2. 行动发生器，它根据(12.34)式产生一个改进策略(即一系列行动)。

3. cost-to-go逼近器，它对状态 i 和参数向量 $\mathbf{w}$ 产生在式(12.33)和式(12.34)中使用的逼近cost-to-go函数 $\hat{J}^{\mu}(i,\mathbf{w})$。

4. 最小二乘求解器，它利用由模拟器对策略 μ 和状态 i 提供的cost-to-go函数 $J^{\mu}(i)$ 的样本，计算使式(12.32)的代价函数最小化的参数向量 $\mathbf{w}$。只有充分评估一个策略和确定一个最优参数向量 $\mathbf{w}^*$ 之后，才能启动从最小二乘求解器到cost-to-go逼近器的连接。此时，由 $\hat{J}^{\mu}(i,\mathbf{w}^*)$ 替代cost-to-go逼近 $\hat{J}^{\mu}(i,\mathbf{w})$。

表12-3给出逼近策略迭代算法的小结。

表12-3 逼近策略迭代算法

已知参数：转移概率 $p_{ij}(a)$ 和代价函数 $g(i,a,j)$。

计算：

1. 选择一个稳定策略 μ 作为初始策略。

2. 使用由模拟器产生的cost-to-go函数 $J^{\mu}(i)$ 的样本集 $\{k(i,m)\}_{m=1}^{M(i)}$，确定神经网络用作最小二乘求解器

$$\mathbf{w}^*=\min_{\mathbf{w}}\mathscr{E}(\mathbf{w})=\min_{\mathbf{w}}\sum_{i\in\mathscr{X}}\sum_{m=1}^{M(i)}(k(i,m)-\hat{J}^{\mu}(j,\mathbf{w}))^2$$

的参数向量 $\mathbf{w}$。

3. 根据第2步决定的参数向量 $\mathbf{w}$，对访问的状态计算逼近cost-to-go函数 $\hat{J}^{\mu}(i,\mathbf{w})$。确定逼近Q-因子：

$$Q(i,a,\mathbf{w})=\sum_{j\in\mathscr{X}}p_{ij}(a)(g(i,a,j)+\gamma\hat{J}^{\mu}(j,\mathbf{w}))$$

4. 确定改进策略

621

$$\mu(i)=\arg\min_{a\in\mathcal{A}_i} Q(i,a,\mathbf{w})$$

5. 重复第2步至第4步

注：第3步和第4步仅在实际访问的状态而不是所有状态应用行动。

很自然，这个算法的运行会有误差，这归因于模拟器和最小二乘求解器的设计不可避免地不完善。对期望的cost-to-go函数进行最小二乘逼近的神经网络可能缺乏适当的计算能力，因而成为第一个误差源。神经网络逼近器的最优化和由此而来的参数向量 $\mathbf{w}$ 的调整是基于

模拟器提供的期望反应，因此成为第二个误差源。假设所有的策略求值和策略改进是分别在 ϵ 和 δ 一定的误差容许限度内完成的，在 Bertsekas and Tsitsiklis(1996)中证明逼近策略迭代算法所产生的策略和最优策略的性能之间差异的因子随 ϵ 和 δ 降低而趋于零。换句话说，逼近策略算法具有最小性能(差异)的可靠保证。根据 Bertsekas and Tsitsiklis(1996)，逼近策略迭代算法初始阶段能够取得迅速而且十分单调的进展，但在极限情况下一个随机性的持续的策略振荡可能发生。这种振荡行为出现在逼近 cost-to-go 函数 $\hat{J}$ 到达最优值 J^* 的区域 $O((\delta+2\gamma\epsilon)/(1-\gamma)^2)$ 内之后，其中 γ 为折扣参数。对所有逼近策略迭代的变体，它们都明显地有导致振荡行为的根本结构。 [622]

12.8 Q－学习

图 12-1 中增强式学习系统的行为目标，是在试验各种可能的行动序列和观察引起的代价以及发生的状态转移之后，如何寻找最优(即最小化代价)策略。在这种背景下我们可能提出下列问题：是否存在仅通过基于形式为

$$s_n = (i_n, a_n, j_n, g_n) \tag{12.35}$$

的样本获得的经验学习最优策略的在线程序？上式中 n 表示离散时间，每个样本 s_n 组成一个四元组，描述为在状态 i 上的一个试验行动 a_n，以代价 $g_n = g(i_n, a_n, j_n)$ 导致对 $j_n = i_{n+1}$ 的状态转移。对于这个基本问题的回答是断然地肯定，它是由 Watkins(1989)发现的一种称为 Q－学习[4]的随机方法。Q－学习是一种增量式的动态规划过程，用一步一步的方式决定最优策略。它非常适合于求解没有明显的转移概率知识的 Markov 决策问题。但是成功应用 Q－学习的关键在于假设环境状态是*完全可观察的*，这就意味着环境是完全可观察的 Markov 链。

回忆 12.4 节中状态－行动对(i, a)的 Q－因子 $Q(i, a)$由式(12.23)定义，而 Bellman 最优性方程由式(12.22)定义。联合这两个方程并且利用(12.20)给出的立即期望代价 $c(i, a)$的定义，我们得到

$$Q^*(i,a) = \sum_{j=1}^{N} p_{ij}(a)\left(g(i,a,j) + \gamma \min_{b\in\mathscr{A}_j} Q^*(j,b)\right) \qquad \text{对所有}(i,a) \tag{12.36}$$

这可看作 Bellman 最优性方程的两步形式。式(12.36)的线性方程组的解对所有状态－行动对(i,a)惟一地定义最优 Q－因子 $Q^*(i,a)$。

我们可以利用基于 Q－因子构造的值迭代算法求解这个线性方程组。因此，对于算法的一步迭代我们有 [623]

$$Q(i,a) = \sum_{j=1}^{N} p_{ij}(a)\left(g(i,a,j) + \gamma \min_{b\in\mathscr{A}_j} Q(j,b)\right) \qquad \text{对所有}(i,a)$$

这个迭代的小步长的形式可描述为

$$Q(i,a) = (1-\eta)Q(i,a) + \eta \sum_{j=1}^{N} p_{ij}(a)\left(g(i,a,j) + \gamma \min_{b\in\mathscr{A}_j} Q(j,b)\right) \qquad \text{对所有}(i,a) \tag{12.37}$$

其中 η 为很小的*学习率参数*，位于区间 $0<\eta<1$ 内。

从它的形式上看，由(12.37)描述的值迭代算法的一次迭代要求转移概率的知识。我们可以构造(12.37)的随机方式从而消除对这一先验知识的需求。特别，在(12.37)的一次迭代

中对所有可能状态求平均被单个样本所替代，因而导出下列对Q-因子的更新公式：

$$Q_{n+1}(i,a) = (1-\eta_n(i,a))Q_n(i,a) + \eta_n(i,a)[g(i,a,j) + \gamma J_n(j)] \quad 对(i,a) = (i_n,a_n) \tag{12.38}$$

其中
$$J_n(j) = \min_{b \in \mathscr{A}_j} Q_n(j,\ b) \tag{12.39}$$

且 j 为后继状态，$\eta_n(i,a)$为在在时间步 n 时状态-行动对(i,a)的学习率参数。更新公式(12.38)应用于当前状态-行动对(i_n,a_n)，根据式(12.35)此时 $j=j_n$。对允许的其余状态-行动对，Q-因子仍保持不变，表示为

$$Q_{n+1}(i,a) = Q_n(i,a) \qquad 对所有(i,a) \neq (i_n,a_n) \tag{12.40}$$

式(13.38)至式(12.40)组成Q-学习算法的一次迭代。

收敛定理

假设学习率参数 $\eta_n(i,a)$满足条件

$$\sum_{n=0}^{\infty}\eta_n(i,a) = \infty \ 和 \sum_{n=0}^{\infty}\eta_n^2(i,a) < \infty \ 对所有(i,a) \tag{12.41}$$

当迭代步数 n 趋于无穷大时，假定所有的状态-行动对被无限地经常访问，那么，对所有状态行动对(i,a)由Q-学习算法产生的Q-因子序列$\{Q_n(i,a)\}$以概率1收敛于最优值$Q^*(i,a)$。

一个保证算法收敛的时变学习率参数的样本为

$$\eta_n = \frac{\alpha}{\beta + n}, n = 1,2,\cdots \tag{12.42}$$

其中 α 和 β 为正数。

总而言之，Q-学习算法是值迭代策略的随机逼近形式，在算法的每一步迭代中它支持单个状态-行动对的Q-因子，即观察到的当前状态和实际执行的行动。最重要的是，无需形成固有的Markov决策过程的明显模型，算法的极限收敛到最优Q-值。一旦最优Q-值可用，利用式(12.30)以相当少的计算便可决定一个最优策略。

Q-学习到最优策略的收敛假设使用Q-因子 $Q_n(i,a)$的查表法表示。这种表示方法简单且计算效率高。但是当由状态-行动对组成输入空间很大或者输入变量是连续的，使用查表法需要大量内存，因而开销特别大。在这种情况下，我们可以利用神经网络进行函数逼近。

逼近Q-学习

式(12.38)和式(12.39)定义当前状态-行动对的Q-因子更新公式。这一对公式可以重写成等价形式

$$\begin{aligned} Q_{n+1}(i_n,a_n) = &\ Q_n(i_n,a_n) \\ &+ \eta_n(i_n,a_n)[g(i_n,a_n,j_n) + \gamma \min_{b \in \mathscr{A}_{j_n}} Q_n(j_n,b) - Q_n(i_n,a_n)] \end{aligned} \tag{12.43}$$

将式(12.43)右边方括号内的表达式当作更新当前Q-因子 $Q_n(i_n,a_n)$的误差信号，我们可以在时间步 n 时确定目标(期望)Q-因子为：

$$Q_n^{\text{target}}(i_n, a_n) = g(i_n, a_n, j_n) + \gamma \min_{b \in \mathcal{A}_{j_n}} Q_n(j_n, b) \tag{12.44}$$

其中 $j_n = i_{n+1}$ 为后继状态。式(12.44)表明在决定目标 Q－因子时后继状态 j_n 发挥关键作用。利用这个目标 Q－因子的定义，我们可以重新构造 Q－学习算法的公式为

$$Q_{n+1}(i,a) = Q_n(i,a) + \Delta Q_n(i,a) \tag{12.45}$$

其中当前 Q－因子的增量改变定义为

$$\Delta Q_n(i,a) = \begin{cases} \eta_n(Q_n^{\text{target}}(i,a) - Q_n(i,a)) & 对(i,a) = (i_n, a_n) \\ 0 & 否则 \end{cases} \tag{12.46}$$

由定义，当前状态 i_n 的"最优"行动 a_n 是在时间步 n 时对该状态具有最小 Q－因子的行动。因此，在状态 i_n 处给定所有允许的行动 $a \in \mathcal{A}_{i_n}$ 的 Q－因子 $Q_n(i_n, a)$，式(12.44)中使用的最优行动 a_n 由下式给出：

$$Q_n = \min_{a \in \mathcal{A}_{i_n}} Q_n(i_n, a) \tag{12.47}$$

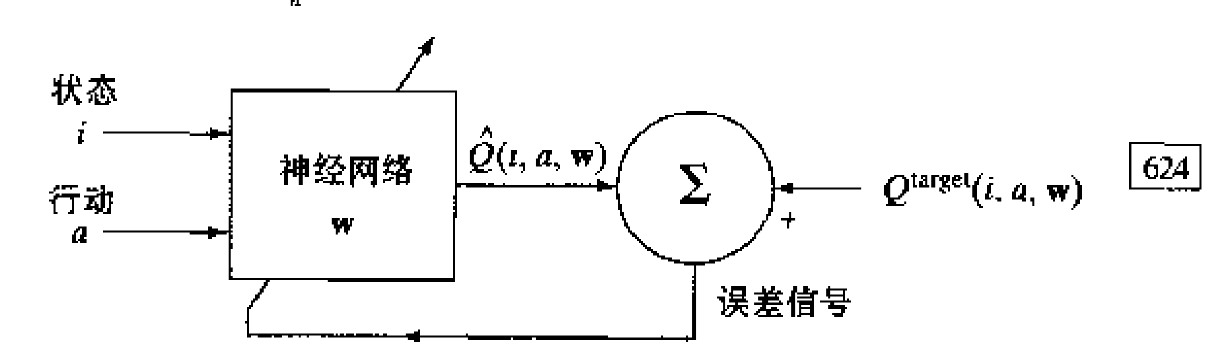


图 12-9 用于逼近目标 Q－因子 $Q^{\text{target}}(i, a, \mathbf{w})$的神经网络设计

624

令 $\hat{Q}_n(i_n, a_n, \mathbf{w})$表示由神经网络(例如利用反向传播算法训练的多层感知器)计算的 Q－因子 $Q_n(i_n, a_n)$的逼近。具有参数向量 **w** 的神经网络的输入为当前状态－行动对(i_n, a_n)，产生输出 $\hat{Q}_n(i_n, a_n, \mathbf{w})$，如图 12-9 所示。在算法的每步迭代中，轻微地改变神经网络的权值向量 **w** 使得输出 $\hat{Q}_n(i_n, a_n, \mathbf{w})$更靠近目标值 $Q_n^{\text{target}}(i_n, a_n)$。但是，一旦权值向量 **w** 改变了，目标值就间接受到影响，也就是改变了值 $Q_n^{\text{target}}(i_n, a_n, \mathbf{w})$。因此不能保证每次迭代都缩短这两个 Q－值间的距离。这也是为什么逼近 Q－学习算法可能发散的原因。如果算法不发散，权值向量 **w** 提供在训练后的神经网络中存储逼近的 Q－因子的手段，因为神经网络输出 $\hat{Q}_n(i_n, a_n, \mathbf{w})$作为对输入$(i_n, a_n)$的响应。

表 12-4 给出逼近 Q－学习算法的小结。

表 12-4 逼近 Q－学习算法小结

1. 从初始权值向量 $\mathbf{w}_0$ 开始，得到 Q－因子 $Q(i_0, a_0, \mathbf{w}_0)$；权值向量 $\mathbf{w}_0$ 借助所用的神经网络完成逼近。
2. 对迭代 $n=1, 2, \cdots$，做下面几步：

(a)对于神经网络设定的 w，确定最优行动

$$a_n = \min_{a \in \mathcal{A}_{i_n}} Q_n(i_n, a, \mathbf{w})$$

(b)确定目标 Q－因子

$$Q_n^{\text{target}}(i_n, a_n, \mathbf{w}) = g(i_n, a_n, j_n) + \gamma \min_{b \in \mathcal{A}_{j_n}} Q_n(j_n, b, \mathbf{w})$$

(c)更新 Q－因子

$$Q_{n+1}(i_n, a_n, \mathbf{w}) = Q_n(i_n, a_n, \mathbf{w}) + \Delta Q_n(i_n, a_n, \mathbf{w})$$

其中

$$\Delta Q_n(i_n, a_n, \mathbf{w}) = \begin{cases} \eta_n(i_n, a_n)(Q_n^{\text{target}}(i_n, a_n, \mathbf{w}) - Q_n(i_n, a_n, \mathbf{w})), & (i,a) = (i_n, a_n) \\ 0, & 其他 \end{cases}$$

(d)应用(i_n, a_n)作为神经网络的输入，产生输出 $\hat{Q}_n(i_n, a_n, \mathbf{w})$作为目标 Q－因子 $Q_n^{\text{target}}(i_n, a_n, \mathbf{w})$的逼近。轻微地改变权值向量使得 $\hat{Q}_n(i_n, a_n, \mathbf{w})$更靠近目标值 $Q_n^{\text{target}}(i_n, a_n, \mathbf{w})$。

(e)回到步骤(a)，重复计算。

探测

在策略迭代中，状态空间的所有潜在重要的部分都应探测到。在Q-学习中我们有一个附加要求：所有潜在有用的行动也都应被测试。特别，对所有允许的状态-行动对应该经常探测足够的次数以满足收敛定理。对于记为μ的贪心策略，只有状态-行动对$(i,\mu(i))$被探测。遗憾的是并不能保证测试所有有用的行动，即使探测完所有状态空间亦是如此。

我们需要的策略是提供两个冲突目标之间的折衷，以此扩展Q-学习(Thrun,1992)：

- 探测，它保证对所有允许的状态-行动对探测足够次数以满足Q-学习收敛定理。
- 利用，它遵循贪心策略以寻求最小化cost-to-go函数。

达到这种折衷的一种方法为遵循混合非稳定(mixed nonstationary)策略，这个策略在一个辅助Markov过程和由Q-学习确定的稳定贪心策略控制的原始Markov过程之间转换(Cybenko,1995)。辅助过程有下列解释：可能状态间的转移概率由原始控制过程的转移概率确定，原始过程具有附加成分，其对应的行动是一致随机性的。混合策略从辅助过程的任何状态开始，随之选择行动，然后切换到原始控制过程，以图12-10中的方式向前或向后进行。消耗在辅助过程上的操作时间占有固定数目的L步，比如说，定义为访问辅助过程所有状态的最长期望时间的两倍。消耗在原始控制过程的时间随每次切换逐步增加。令n_k表示从辅助过程到原始控制过程的切换时间，m_k表示切换回辅助过程的时间，n_k和m_k分别定义为

$$n_k = m_{k-1} + L, k = 1,2,\cdots, \quad 且 \quad m_0 = 1$$

和

$$m_k = n_k + kL, \quad k = 1, 2, \cdots$$

构造辅助过程使得当$k\to\infty$时，以概率1访问所有状态无穷次，因而保证收敛到最优Q-因子。进一步，当$k\to\infty$，混合策略在辅助过程上所消耗的操作时间渐进地为消耗在原始控制过程的操作时间的一小部分，这就意味着混合策略渐进收敛到一个贪心策略。因此，如果Q-因子收敛到它们的最优值，贪心策略确实必定是最优的，只要策略变为贪心策略时足够地慢。

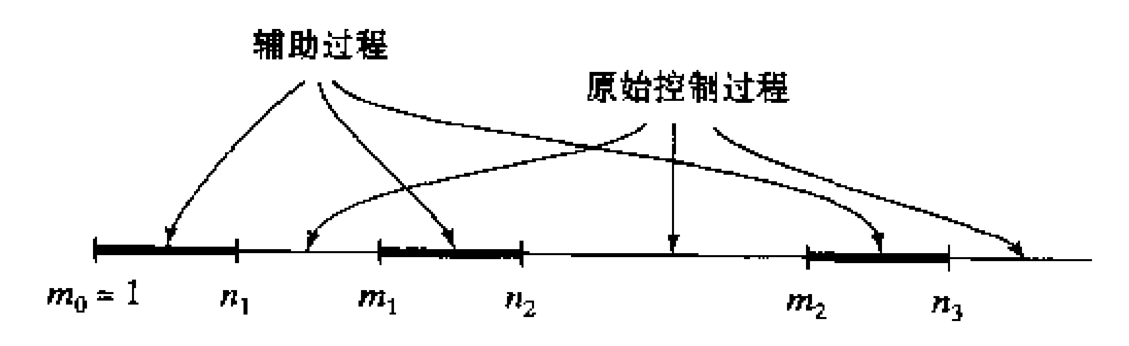


图12-10 属于辅助过程和原始控制过程的时间段

12.9 计算机实验

在这个计算机实验中重新讨论在例12.1中考虑的驿车问题。这次我们利用逼近Q-学习求解问题。利用两种方法实现算法：一种方法使用表来表示Q-值，另一种方法使用神经网络。

图12-11给出使用表方法的下列Q-因子的学习历史：$Q(A,\text{up})$，$Q(C,\text{straight})$，$Q(E,\text{straight})$和$Q(I,\text{up})$。在图12-11中虚线表示期望的Q-值。每次试验为从状态I到目

标状态 J 的完整路线，每次试验的开始状态随机挑选，学习率参数 $\eta_n(i,a)$ 定义为

$$\eta_n(i,a) = \frac{\alpha v_n(i,a)}{K + v_n(i,a)}$$

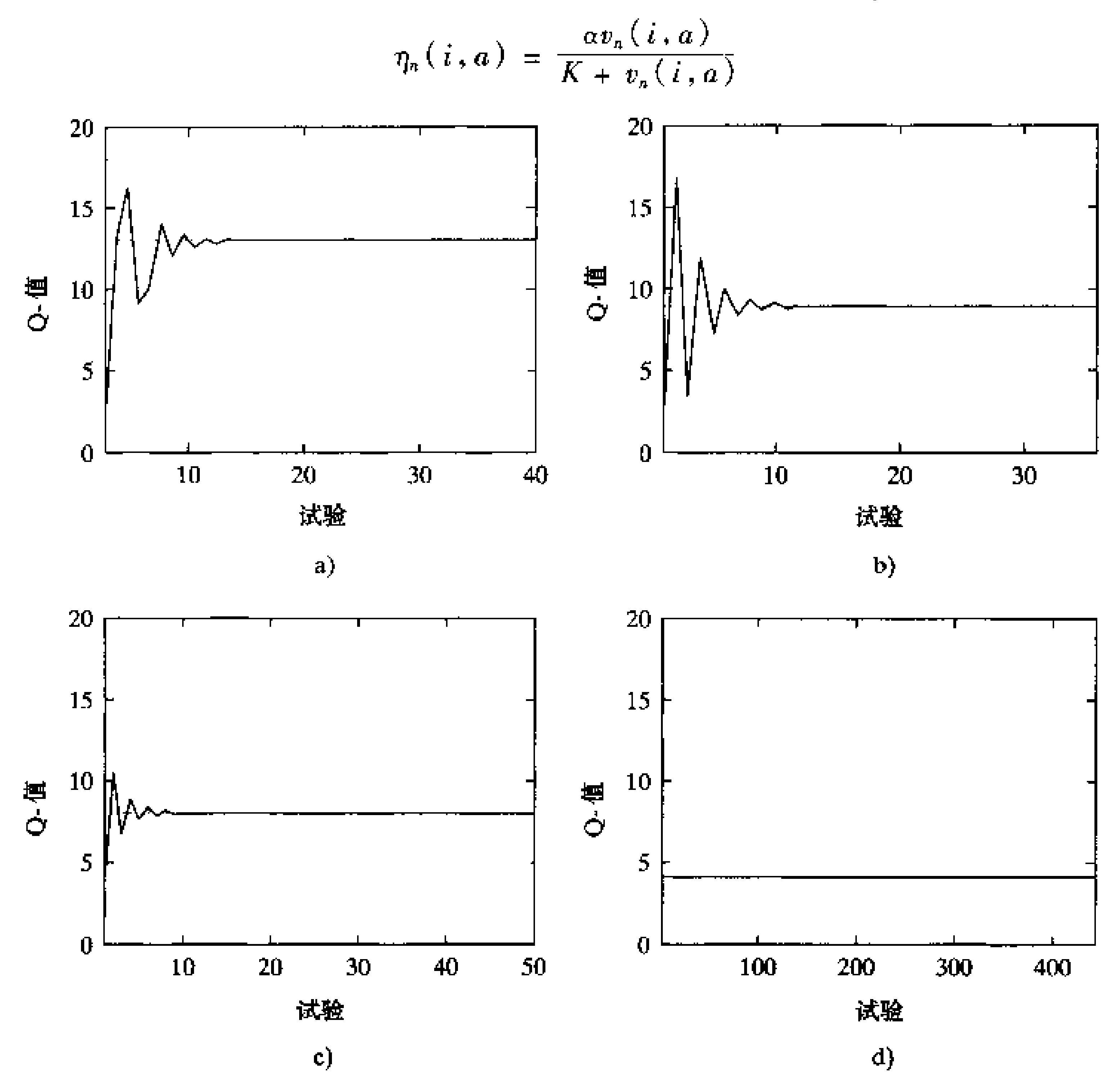


图 12-11 利用查表法求解驿车问题的学习曲线

a) $Q(A,\text{up})$的学习曲线 b) $Q(C,\text{straight})$的学习曲线

c) $Q(E,\text{straight})$的学习曲线 d) $Q(I,\text{up})$的学习曲线

其中 $v_n(i,a)$为当前时刻 n 为止所访问的状态－行动对的数目，$\alpha=1.6$，$K=600$。总共完成1000次之后，找到最优路线为

$$A \to D \to F \to I \to J$$

这是一条确认为最优路线，总的代价为 11。

图 12-12 表示利用两个输入节点、10 个隐藏单元和 1 个输出神经元的多层感知器获得的相应结果。一个输入节点代表状态而另一个节点代表从一个状态到下一个所采取的行动。多层感知器的输出表示网络计算出的 Q－值。网络使用标准的反向传播算法。在时刻 n 时使用的目标 Q－值利用(12.44)计算。学习率参数设置为 0.012，没有使用动量。对每个状态－行动对训练网络 10 000 次。图 12-12 表示 Q－值的学习历史：$Q(A,\text{up})$，$Q(C,\text{straight})$，$Q(E,\text{straight})$和 $Q(I,\text{up})$。网络发现的最优路线为

$$A \to D \to E \to H \to J$$

627

这仍是一条被承认的最优路线，总代价为 11。

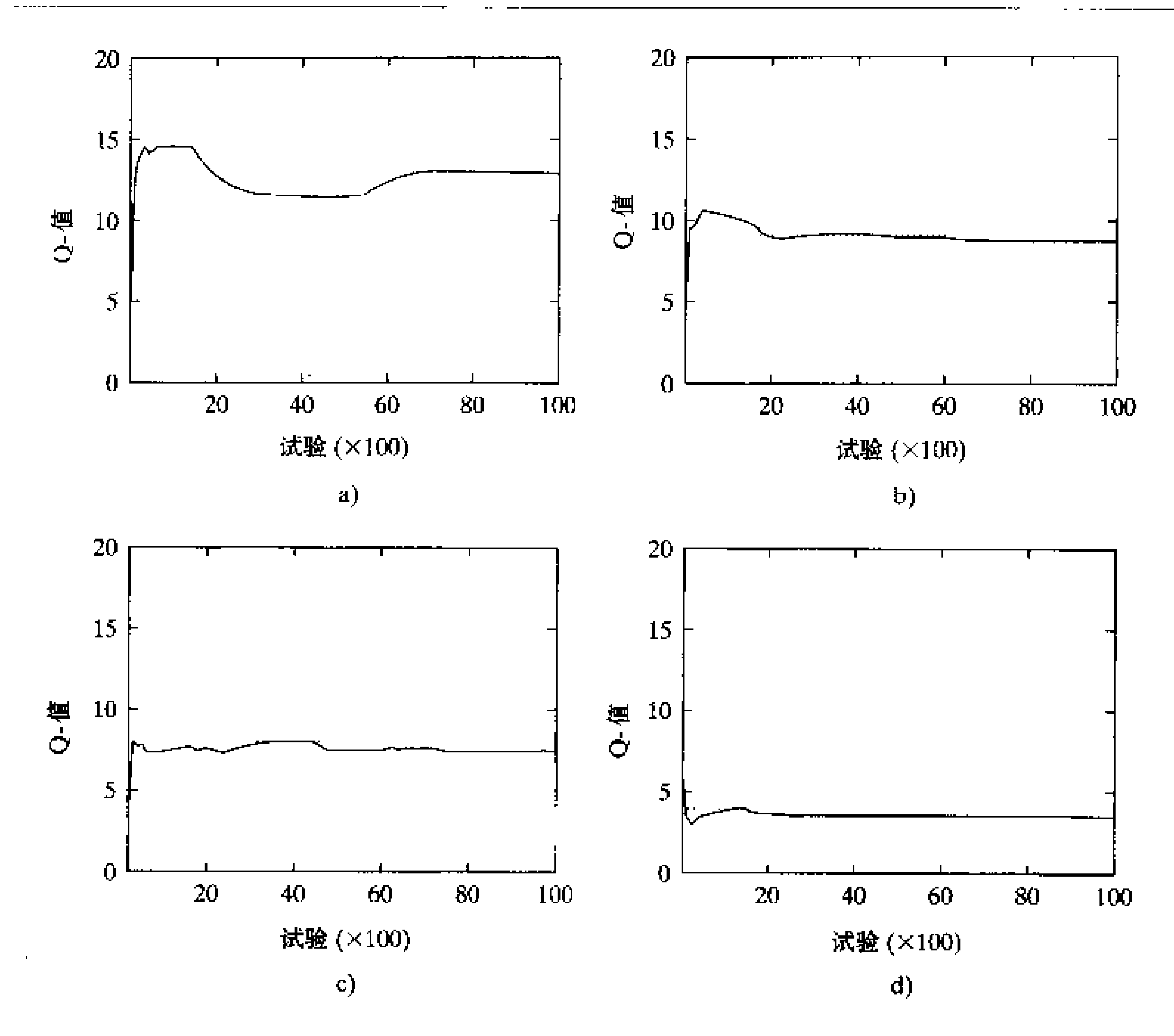


图 12-12 利用神经网络求解驿车问题的学习曲线

a) $Q(A,\mathrm{up})$的学习曲线 b) $Q(C,\mathrm{straight})$的学习曲线

c) $Q(E,\mathrm{straight})$的学习曲线 d) $Q(I,\mathrm{up})$的学习曲线

两种实现方法的计算要求小结如下：

(a)神经网络：

输入数目 = 2

隐藏神经元数目 = 10

输出神经元数目 = 1

突触权重和偏置总数目 $= 2\times 10 + 10 + 10\times 1 + 1 = 41$

(b)查表法：

状态数目 = 10

行动数目 = 2 或 3

表格大小 = 21

628 在这个实验中可能的状态数目很小，导致的结果是查表法比神网络要求更少的存储。但是在大规模问题中状态数目非常大，神经网络常常在存储要求方面比查表法获得优势。

12.10 小结和讨论

结合经典的动态规划的数学形式和神经网络的学习能力，神经动态规划为需要规划的行

为任务提供强有力的求解方法。在增强式学习这个现代方法中，系统学会做两件事：通过观察它自身的行为做出好的决策，和通过增强机制改进它的行动。固有的决策过程服从 Markov 模型。

在本章我们描述了两种神经动态规划过程：

1. 逼近策略迭代。策略迭代在两个基本步骤之间交替：

- 策略求值，确定当前策略的 cost-to-go 函数。
- 策略改进，对当前 cost-to-go 函数用贪心策略更新当前策略。

在逼近策略迭代中，结合模拟和函数逼近以评估策略。为了模拟系统的 Markov 模型，要求知道状态转移概率。为了进行函数逼近，我们可以利用神经网络(例如多层感知器、径向基函数网络或支持向量机)，由于它的通用逼近性质，这是比较适合的。 629

2. 逼近 Q－学习。在值迭代中，作为策略迭代的替代物，利用收敛于最优策略的逐次逼近过程求解 Markov 决策问题。Q－学习是值迭代的异步形式，这是为了避免需要状态转移概率的明显知识而构造的。它具有如下富有吸引力的性质：

- 如果所有的状态－行动对都被无限经常地访问，且学习率参数满足由式(12.41)给定的条件，那么 Q－学习以概率 1 收敛到最优 Q－因子。
- Q－学习直接更新和最优策略相关的 Q－因子估计，从而避免策略迭代中涉及的多次策略求值步骤。

在逼近 Q－学习中，利用神经网络逼近 Q－因子的估计是为了在可能的状态数目很大时避免需要过量的存储要求。简言之，逼近 Q－学习是在无系统模型可用且存储要求过大的情况下用于求解 Markov 决策问题的基于模拟的算法。当然，它甚至可用于有系统模型可用的情况，这时它提供逼近策略迭代的一种替代。

神经动态规划技术在求解主要关心的规划为大规模问题时有特殊的效果。对于这类问题，由于需要搜索的状态空间太大，传统的动态规划方法很难应用。确实，神经动态规划已成功应用于求解许多不同领域的困难的现实世界的问题，包括十五子棋(Tesauro, 1989, 1994)，组合优化(Bertsekas and Tsitsiklis, 1996)，电梯调度(Crites and Barto, 1996)和动态频段分配(Singh and Bertsekas, 1997; Nie and Haykin, 1996, 1998)。下面我们稍微详细地描述对十五子棋的应用。

在 Tesauro(1994)首次报告了基于神经网络的计算机程序选手玩十五子棋，随后在 Tesauro (1994)给出了改进，它是一个给人印象特别深刻的成功故事，并且已成为推动神经动态规划中研究的源泉。十五子棋是一种古老的双人棋盘游戏。沿着一条有效的一维路径对弈。游戏者双方轮流掷一对骰子，相应地沿路径的相反方向移动他们的棋子。游戏者的合法移动棋子依赖于掷骰子的结果和棋盘布局。首先把自己的所有棋子移到棋盘的最终目标者为胜者。游戏可用一个 Markov 决策过程建模。它的状态定义为棋盘布局的描述、掷骰子的结果和游戏者作的移动。Tesauro(1989)利用监督学习建立了神经－十五子棋的最初形式。给定状态的“初始”描述，它能学会中等以上的水平。报道中也许最有趣的发现为良好的规模效应，也就是说，神经网络的大小和训练次数增加到一定规模，可以观察到性能有重要的提高。研究使用的神经网络为使用反向传播算法训练的多层感知器(MLP)。利用具有 40 个隐藏神经元的 MLP 对总共 200 000 局游戏进行训练获得了最好的性能。在随后的 Tesauro(1994)研究报告中，利用一种称为乐观(optimistic) TD(λ)的策略迭代形式训练神经网络。TD(λ)代表时序差分学 630

习，这归功于Sutton(1988)。乐观TD(λ)是用于逼近cost-to-go函数J^μ的基于模拟的方法，在该方法中策略μ被新的策略μ所替代，新策略μ在每步状态转移时逼近J^μ是贪心的(Bertsekas and Tsitsiklis,1996)。基于这个神经动态规划方法的计算机程序通常称为TD－十五子棋；Tesauro添加了状态的(即特征)提取函数作为神经网络输入表示，使得TD－十五子棋达到优秀大师的水平，非常接近于世界上最好的棋手。支持这个论断的事例是大量有关TD－十五子棋和几个世界级棋王进行对弈的试验(Tesauro,1995)。

注释和参考文献

[1] 增强式学习的传统处理方法植根于心理学，可追溯到Thorndike(1911)关于动物学习早期的工作和Pavlov(1927)关于条件反射的研究。对传统增强式学习的方法的贡献还包括Widrow et al.(1973)的工作；在那篇文章中，引入了评价(critic)的概念。Hampson(1990)以书的形式讨论传统的增强式学习。

对现代增强式学习的主要贡献包括Samuel(1959)有关他的著名的棋子游戏程序的工作，Barto et al.(1983)关于自适应评价系统的工作，Sutton(1988)关于时序差分(temporal difference)方法的工作和Watkins(1989)关于Q－学习的工作。White and Sofge(1992)关于智能控制的手册给出关于White和Jordan的最优控制、Barto的增强式学习和自适应评价方法以及Werbos的启发式动态规划的材料。

Bertsekas and Tsitsiklis(1996)第一次以书的形式给出现代增强式学习的处理。有关增强式学习的历史资料，参看Sutton and Barto(1998)。

[2] 动态规划由R. E. Bellman等在20世纪50年代晚期提出，参看Bellman(1957)，Bellman and Dreyfus(1962)，有关该主题的详细展开参看Bertsekas(1995b)的两卷书。

[3] 策略迭代和值迭代是动态规划的两个主要方法。另外有两个值得注意的方法：Gauss-Seidel方法和异步动态规划(Barto et al.，1995；Bertsekas，1995b)。在Gauss-Seidel方法中，串行扫描所有状态，每个状态根据其他状态的最新代价进行竞争，在一个时刻只更新一个状态的cost-to-go函数。异步动态规划和Gauss-seidel的区别在于它没有组织成系统化的依次扫描状态集。

[4] Watkin(1989)在他的博士论文的第96页，对Q－学习做如下评语：

"附录1给出这个学习方法对有限Markov决策过程工作的证明。证明也表明该学习方法会很快收敛到最优行动－值函数。虽然这是非常简单的思想，据我所知，以前从未被明显提出。但是必须指出，有限Markov决策过程和随机动态规划用于若干不同领域已经被广泛研究三十多年了，它不像Monte－Carlo方法那样以前无人考虑过。"

631

在对这些评论的一个足注中，Barto et al.(1995)指出，虽然对状态－行动对赋值的思想被Denardo(1967)所采用，构成动态规划方法的基础，但他们没有看见比Watkins的1989论文更早的像Q－学习这样用于估计这些值的算法。

[5] Watkins(1989)给出Q－学习收敛定理证明的概要，后来在Watkins and Dayan(1992)中给出了其改进。Tsitsiklis(1994)给出了Q－学习收敛的更一般的结果，也可参考Bertsekas and Tsitsiklis(1996)。

习题

Bellman 的最优准则

12.1 当折扣因子 γ 接近于 1 时，(12.22)中 cost-to-go 函数的计算变长。为什么？说明你的回答的理由。

12.2 在本题中我们给出由 Ross(1983)得到的关于 Bellman 最优性方程(12.22)的另一个证明。

(a)令 π 为任意策略，假设 π 在时间步 0 选择行动 a 的概率为 p_a，$a\in\mathscr{A}_i$。那么

$$J^{\pi}(i)=\sum_{a\in\mathscr{A}}p_a\Big(c(i,a)+\sum_{j=1}^{N}p_{ij}(a)W^{\pi}(j)\Big)$$

其中 $W^{\pi}(j)$代表从时间步 1 以前的 cost-to-go 函数的期望，这里假设在时间步 1 状态为 j 且使用策略 π。由此证明

$$J^{\pi}(i)\geqslant\min_{a\in\mathscr{A}_i}\Big(c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J(j)\Big)$$

其中

$$W^{\pi}(j)\geqslant\gamma J(j)$$

(b)令 π 是在时间步 0 选择行动 a_0 的策略，如果下一个状态为 j，可看作过程以状态 j 开始，遵循策略 π_j 使得

$$J^{\pi_j}(j)\leqslant J(j)+\varepsilon$$

其中 ε 是一很小正数。由此证明

$$J(i)\geqslant\min_{a\in\mathscr{A}_i}\Big(c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J(j)\Big)+\gamma\varepsilon$$

(c)用(a)和(b)导出的结果证明式(12.22)。

12.3 式(12.22)表示 N 个方程的线性方程组，每个状态用一个方程。令

$$\mathbf{J}^{\mu}=[J^{\mu}(1),J^{\mu}(2),\cdots,J^{\mu}(N)]^T$$

$$\mathbf{c}(\mu)=[c(1,\mu),c(2,\mu),\cdots,c(N,\mu)]^T$$

$$\mathbf{P}(\mu)=\begin{bmatrix}p_{11}(\mu) & p_{12}(\mu) & \cdots & p_{1N}(\mu)\\ p_{21}(\mu) & p_{22}(\mu) & \cdots & p_{2N}(\mu)\\ \vdots & \vdots & & \vdots\\ p_{N1}(\mu) & p_{N2}(\mu) & \cdots & p_{NN}(\mu)\end{bmatrix}$$

证明式(12.22)可以重新写成等价的矩阵形式：

$$(\mathbf{I}-\gamma\mathbf{P}(\mu))\mathbf{J}^{\mu}=\mathbf{c}(\mu)$$

其中 $\mathbf{I}$ 为单位矩阵。讨论表示 N 个状态的 cost-to-go 函数的向量 $\mathbf{J}^{\mu}$ 的惟一性。

12.4 在 12.3 节中我们推导用于有限范围问题的动态规划算法。在本题中对一个折扣问题重新推导这个算法，其中 cost-to-go 函数由下式定义：

$$J^{\mu}(X_0)=\lim_{K\to\infty}\Big[\sum_{n=0}^{K-1}\gamma^n g(X_n,\mu(X_n),X_{n+1})\Big]$$

特别地，证明

$$J_K(X_0)=\min_{\mu}\underset{X_1}{E}[g(X_0,\mu(X_0),X_1)+\gamma J_{K-1}(X_1)]$$

[632]

策略迭代

12.5 在12.4节中我们说cost-to-go函数满足

$$J^{\mu_{n-1}} \leqslant J^{\mu_n}$$

证明这个论断。

12.6 讨论式(12.25)描述的论断的重要性。

12.7 利用控制器评价系统(controller critic system)，说明策略迭代算法中策略更新和策略求值之间的相互作用。

值迭代

12.8 一个动态规划问题涉及总共N个允许状态M个允许行动。假定使用一个稳定策略，证明值迭代算法的一次迭代需要阶为N^2M的操作。

12.9 表12-2给出依据对状态$i \in \mathscr{X}$的cost-to-go函数$J^{\mu}(i)$构造的值迭代算法公式的小结。依据Q－因子$Q(i, a)$重新构造这个算法公式。

12.10 策略迭代总是在有限步后终止，但是值迭代可能要无限次迭代。讨论这两个动态规划方法之间的其他差异。

Q－学习

12.11 证明

633

$$J^*(i) = \min_{a \in \mathscr{A}_i} Q(i, a)$$

12.12 Q－学习算法有时称作值迭代策略的自适应形式。证明这种描述的正确性。

12.13 构造由表12-4小结的逼近Q－学习算法的信号流图。

12.14 表12-4小结的逼近Q－学习算法假定缺乏状态转移概率的知识。假定可以用这些概率，重构这个算法。

634

第 13 章 使用前馈网络的时序处理

13.1 简介

时间是学习过程的基本组成。它可以是连续的，也可以是离散的。无论其形式如何，时间是一个有序实体，是在实践中遇到的许多认知任务如视觉、语音、信号处理以及马达控制的基础。通过将时间引入神经网络的运行，使它能跟踪在一些非平稳过程(如语音信号、雷达信号、发动机引擎信号、股票市场价格波动)中统计的变化。问题是：我们如何在神经网络运行中嵌入时间？这个基本问题的答案在于两个可能方法之一：

- *隐式表示*。时间是通过其作用于信号处理的效果以一种隐含方式来表示的[1]。例如，输入信号经过统一采样，和网络输入层相连的每个神经元的突触权值序列和输入样本的不同序列作*卷积*(convolved)。这样，输入信号的时间结构嵌入在网络的空间结构里。
- *显式表示*。时间由它自身的特定表示给出[2]。如蝙蝠的回声定位系统是通过发射短的频率调制(FM)信号，使得对于每个限制在 FM 扫描期间的很短的一个时间段的频道维持相同的强度等级。被一组听觉接收器编码的几个不同频率之间的多种比较是为了抽取目标物的准确的距离信息(Suga and Kanwal, 1995)。当从目标的回声在经一段未知时延以后被接收时，一个具有匹配的延迟线的神经元(在听觉系统)进行响应，从而提供目标范围的估计值。

635

在这一章里我们关心时间的隐式表达，这由对一个静态神经网络(如多层感知器)提供动态属性而得到。从而使得神经网络对信息承载信号的时间结构作出响应。

为了使神经网络为动态的，必须给它*记忆*(memory)。正如第 2 章指出那样，记忆可分为“短期”和“长期”记忆，这要依赖于保留时间。神经网络的长期记忆是通过监督学习建立的，由此训练数据集的信息内容存储(部分或者全部)在网络的突触权值上。但是，如果当前的这项任务有一个时间维数，我们需要某种形式的短期记忆使神经网络为动态的。一个简单的在神经网络结构内建立短期记忆的方法就是使用*时延*(time delay)，这可以在网络内部的突触层或网络的输入层来实现。在神经网络中使用时延是受到神经生物学启发，因为在人脑中信号延迟无处不在，并且它在神经生物信息处理中起着重要作用(Braitenberg, 1967, 1977, 1986; Miller, 1987)。

本章的组织

本章的内容分为三个部分。第一部分，包括 13.2 节和 13.3 节，论述网络结构和模型。在 13.2 节，我们讨论记忆的结构，接下来的 13.3 节描述对于信号时间处理的两种不同的网络结构。

本章的第二部分包括 13.4 节到 13.6 节，论述一类被称为集中时滞的前馈网络的神经网络；术语“集中”(focused)指的是短期记忆被全部放置在网络的前端。在 13.6 节讨论这一结

构的计算机实验。

本章第三部分，包括13.7节到13.9节，论述分布式时滞前馈网络，在这种网络中延迟线被分布于整个网络。13.7节描述一个神经元的时空模型，接下来在13.8节论述刚刚提到的第二类神经网络。在13.9节讨论用于分布式时滞前馈网络的监督学习的"时序"反向传播算法。

这一章在13.10节中以一些最后评论作为结束。

13.2 短期记忆结构

记忆的主要作用是将一个静态的网络转变成一个动态的网络。特别地，将记忆嵌入到诸如通常的多层感知器的静态网络结构中，网络的输出变成时间的函数。建立非线性动态系统的这种方法是直接的，因为它对职责作了明确的分离：静态网络负责非线性的处理，而记忆负责时间的相关处理。

636 短期记忆[3]可以在连续的时间或离散的时间中实现。连续时间用 t 表示，离散的时间用 n 表示。图13-1中电阻-电容电路图就是一个连续时间记忆的例子，它的特征是的脉冲响应（即记忆痕迹）$h(t)$按时间 t 的指数函数衰减。在本章后面描述的神经元加性模型的模拟实现中，这个电路在突触级负责记忆。这一节我们主要关心离散时间记忆。

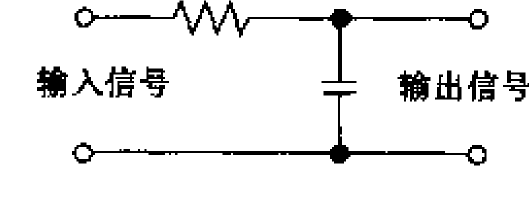


图13-1 电阻电容电器

处理离散时间系统的一个有用工具是 z-变换。令$\{x(n)\}$表示离散时间序列，可以扩展到无限的过去。它的 z-变换 $X(z)$定义为

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{13.1}$$

其中 z^{-1}是单元延迟操作符；也就是说，z^{-1}作用在 $x(n)$上，产生延迟形式 $x(n-1)$。假设 $x(n)$用于脉冲响应 $h(n)$的一个离散时间系统。这个系统的输出 $y(n)$由下面的卷积和定义：

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{13.2}$$

当 $x(n)$等于单元脉冲时，$y(n)$产生系统的脉冲响应 $h(n)$。z-变换的一个重要性质是时间域上的卷积变成 z 域上的乘积(Oppenheim and Schafer, 1989; Haykin and Van Veen, 1998)。我们如果定义序列$\{h(n)\}$和$\{y(n)\}$的 z-变换分别为 $H(z)$和 $Y(z)$，则有

$$Y(z) = H(z)X(z) \tag{13.3}$$

637 或者等价地

$$H(z) = \frac{Y(z)}{X(z)} \tag{13.4}$$

函数 $H(z)$称为该系统的传递函数(transfer function)。

图13-2显示一个含有 p 个相同节点级联的离散时间记忆框图；今后 p 称为记忆的阶。每个延迟片段，可以看作操作符，由传递函数 $G(z)$定义其特征（如图所示）。同样，每个片段可以根据脉冲响应 $g(n)$来描述，具有下述两个特征：

- 它是因果的，即当 $n<0$ 时，$g(n)=0$。
- 它是归一化的，即有 $\sum_{n=0}^{\infty} |g(n)| = 1$。

因此 $g(n)$称为离散时间记忆的产生核。

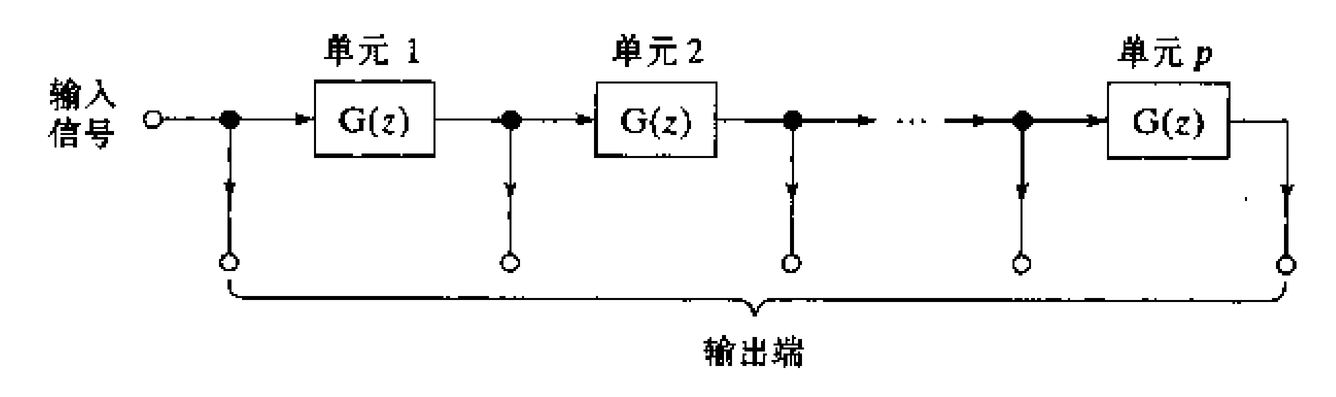


图 13-2 p 阶广义抽头延迟线记忆

根据图 13-2，我们可以形式地定义离散时间记忆为线性时间不变的单输入多输出系统(single input-multiple output，SIMO)，并且其产生核满足上述两个条件。那些与记忆输出端点相连接的节点，称为抽头(tap)。注意对一个 p 阶的记忆来说，共有 $p+1$ 个抽头，只有一个抽头是属于输入。

可以用深度和分辨率来衡量记忆结构的属性。设记忆结构中总的脉冲响应为 $g_p(n)$，定义为 $g(n)$ 的 p 个逐次卷积，或者等价于 $G^p(z)$ 的逆 z - 变换。记忆深度记为 D，定义为 $g_p(n)$ 的第一时间矩(moment)，表示为

$$D = \sum_{n=0}^{\infty} n g_p(n) \tag{13.5}$$

一个低深度 D 的记忆只能将信息内容保持较短的时间，而高深度的记忆则能保持较长时间。记忆分辨率记为 R，指的是每个单位时间内记忆结构中的抽头数目。一个高分辨率 R 的记忆结构能将输入的序列信息保持在精确的层次上，而低分辨率的记忆结构只能保持在粗糙的层次上。当抽头数目固定时，记忆深度和记忆分辨率的乘积对 p 阶记忆是一个常量。

选择不同的产生核 $g_p(n)$ 会产生不同的深度 D 和分辨率 R，这可以用下面两个记忆结构来说明。

抽头延迟线记忆 图 13-3 显示的框图是短期记忆最简单和最常用的形式，称为抽头延迟线记忆(tapped delay line memory)。它包含 p 个单位延迟操作符，每个都表示为 $G(z) = z^{-1}$。也就是说，产生核为 $g(n) = \delta(n-1)$，其中其中 $\delta(n)$ 是单位脉冲

$$\delta(n) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} \tag{13.6}$$

638

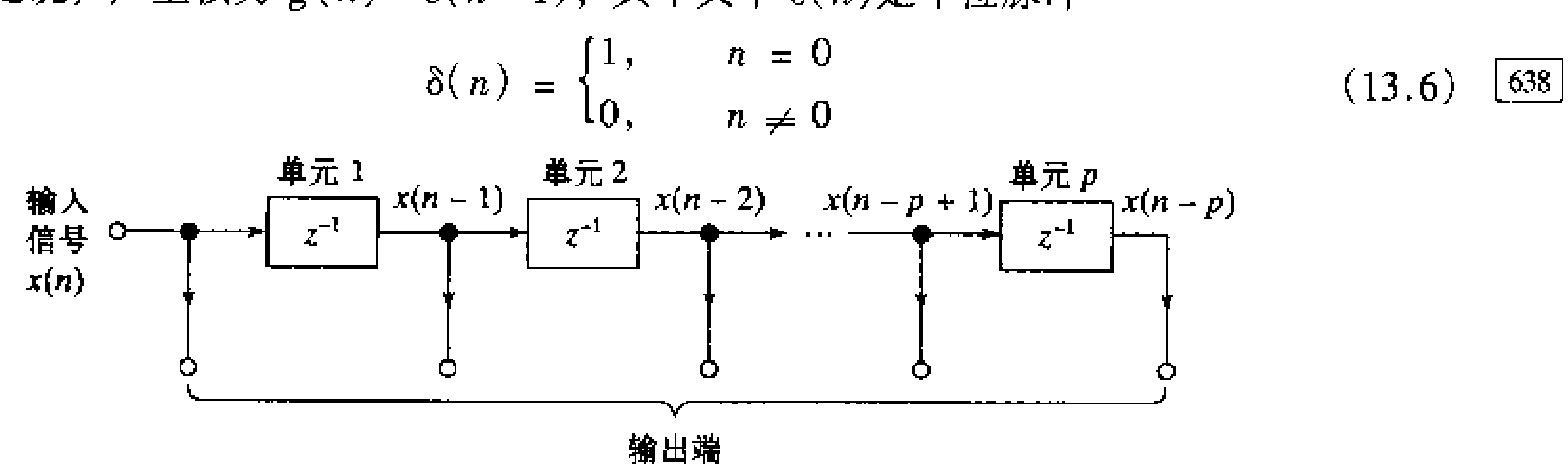


图 13-3 通常的 p 阶抽头延迟线记忆

图 13-3 的抽头延迟线的总脉冲响应为 $g_p(n) = \delta(n-p)$。将 $g_p(n)$ 代入式(13.5)中产生记忆深度 $D = p$，这一点直观上是满足的。从图 13-3 中我们可以看出每个单位时间内只有一个抽头；因此，$R = 1$。这样抽头延迟线的记忆深度随着 p 的阶数增大而线性增长，但是它的记忆深度在单位时间内是固定不变的；并且它的深度 - 分辨率乘积也是一个常数。

我们需要额外的自由度去实现对于记忆深度的控制。这种准备可以通过下面考虑的一个对抽头延迟线的替代来提供。

Gamma记忆 图13-4显示用于被称为gamma记忆的记忆结构的基本功能块 $G(z)$ 的信号流图。特别地，记忆结构的每个部分包含一个带有单位延迟 z^{-1} 的反馈环以及一个可调整的参数 μ。每一个这样部分的传递函数为

图13-4 一个gamma记忆片段的信号流图

$$G(z)=\frac{\mu z^{-1}}{1-(1-\mu)z^{-1}}=\frac{\mu}{z-(1-\mu)} \tag{13.7}$$

为了稳定性，$G(z)$ 在 $z=1-\mu$ 处的惟一极点必须在 z 平面的单位圆内。这就要求

$$0<\mu<2 \tag{13.8}$$

Gamma记忆的产生核是 $G(z)$ 的逆 z－变换，即

$$g(n)=\mu(1-\mu)^{n-1},\quad n\geqslant 1 \tag{13.9}$$

式(13.8)中的条件保证 $g(n)$ 随着 n 增大至无穷而指数地衰减到零。

Gamma记忆总的脉冲响应是总的传递函数的逆 z－变换

$$G_p(z)=\left(\frac{\mu}{z-(1-\mu)}\right)^p$$

639 即

$$g_p(n)=\binom{n-1}{p-1}\mu^p(1-\mu)^{n-p},\qquad n\geqslant p \tag{13.10}$$

其中(:)是由 $\binom{n}{p}=\frac{n(n-1)\cdots(n-p+1)}{p!}$ 定义的二项式系数，n 和 p 为整数。对于不同的 p，总的脉冲响应 $g_p(n)$ 表示Gamma函数的被积函数的离散形式(deVries and Principe,1992)，这正是记忆命名的原因。图13-5显示一簇脉冲响应 $g_p(n)$，它们对 μ 归一化，其中 $\mu=0.7$，$p=1,2,3,4$。注意在图13-5中时间坐标轴按参数 μ 标度。这种标度具有将 $g_p(n)$ 的峰值定位在 $n=p$ 的作用。

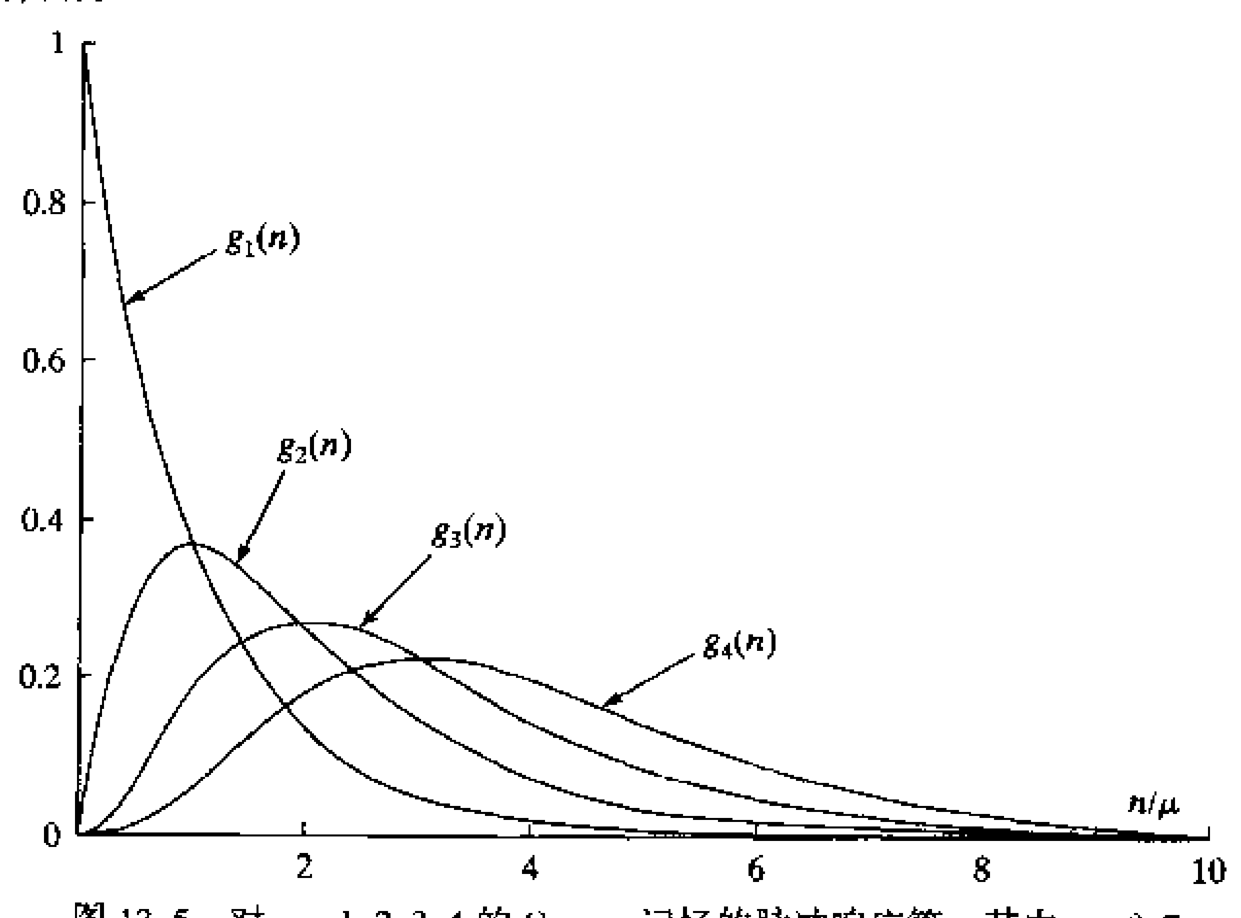


图13-5 对 $p=1,2,3,4$ 的Gamma记忆的脉冲响应簇，其中 $\mu=0.7$

Gamma 记忆的深度为 p/μ，分辨率为 μ，深度-分辨率的乘积为 p。相应地，通过选择小于单位的 μ 值，对于特定的阶 p，Gamma 记忆在抽头延迟线的深度有所提高(但是牺牲了分辨率)。当 $\mu=1$ 时，这些量将减至各自的抽头延迟线上假设的值。因此，抽头延迟线只是 Gamma 记忆的一个特例。这个结论同样可以在式(13.9)中设置 $\mu=1$ 得到证实。如果 μ 大于 1 而小于 2，那么 $(1-\mu)$ 在这个方程中变为负值，但是绝对值小于 1。

13.3 用于时序处理的网络体系结构

时序处理的网络结构不只一种形式，这正如记忆结构一样。在这一节我们将描述两种前馈网络体系结构，它们分别以自己的方式丰富了时序处理文献。

NETtalk

NETtalk 由 Sejnowski and Rosenberg(1987)设计，是将英语语音转化为音素的一个大规模并行分布式网络的一个例子。一个音素(phoneme)是一个基本的语言单位。图 13-6 就显示一个 NETtalk 的示意图，它建立在一个多层感知器的基础上，输入层有 203 个感知节点的，隐藏层有 80 个神经元，输出层有 26 个神经元。所有神经元使用 sigmoid(logistic)型激活函数。这个网络的突触连接有 18 629 个，每个神经元包含有可变的阈值。阈值是偏置的负值。这个网络使用标准的反向传播算法进行训练。 640

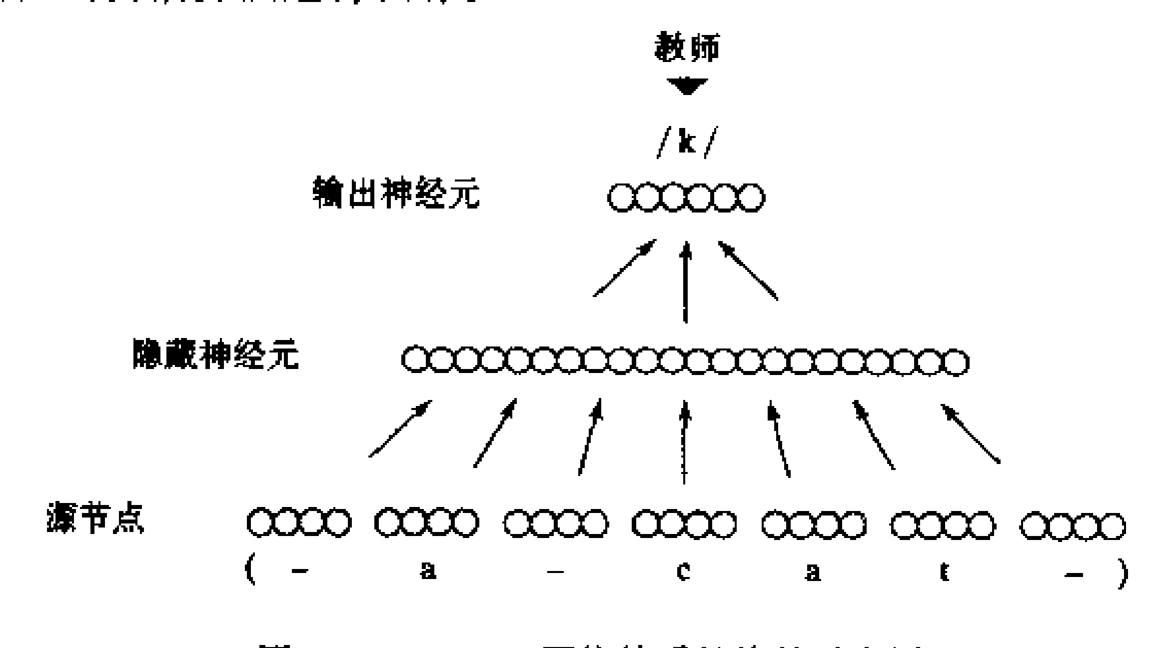


图 13-6 NETtalk 网络体系结构的示意图

这个网络有七组输入节点。每组对输入文本的 1 个字母进行编码。从而每次将 7 个字母组成的串呈现给输入层。训练过程的期望响应是和 7 个字母窗口中央的一个(即第 4 个)相联系的正确音素。另外 6 个字母(在中间字母两边各 3 个)对网络的每一个决策来说提供部分的上下文。通过一个字母接着一个字母的方式使文本通过窗口。在处理的每一步中，网络都计算一个音素，每学完一个单词后，网络的突触权值就根据计算出的发音与正确的发音的接近程度进行调整。

NETtalk 的性能展示了和观察到的人特性的一些相似之处，可总结为以下几点(Sejnowski and Rosenberg, 1987)。

- 训练遵守有力的规律(power law)。
- 网络学习的单词越多，它泛化和新词正确发音的性能就越好。
- 当网络的突触连接受破坏时，网络性能的下降非常缓慢。

• 在网络遭到破坏以后，进行重新学习，学习的速度要比原始(以前的)训练快得多。

NETtalk 出色地说明了学习的很多方面的微小细节，在开始的时候，在它的输入模式中具有大量“先天”的知识并且通过实践逐渐获得将英语语音转化为音素的能力。但是，它还没有走向实际的应用。

时延神经网络

使用普通的时间延迟来执行时序处理的通用神经网络就是所谓的时延神经网络(time 641 delay neural network, TDNN)，由 Lang and Hinton(1988)和 Waibel et al.(1989)第一次描述。TDNN 是一个多层前馈网络，其隐藏层神经元和输出神经元都是沿时间复制。它被设计用于显式地捕获在利用声谱图(spectrogram)识别一个孤立单词(音素)的过程中遇到的时间对称性的概念。一个声谱图是一张两维的图像，其纵轴表示频率，横轴表示时间。图像的强度(灰度)与信号的能量相对应(Rabiner and Schafer, 1978)。图 13-7 显示 TDNN 一个隐藏层形式(Lang and Hinton, 1988)。输入层包括 192(16 × 12)个用于对声谱进行编码的感知节点。隐藏层包含 8 个隐藏神经元的 10 次复制；而输出层包含 4 个输出神经元的 6 次复制。一个隐藏神经元的不同复制应用相同突触权值集合到很窄的(三倍于时间步长)声谱窗口之中；相似地，输出神经元的不同复制应用相同突触权值集合到由隐藏层计算出的伪声谱图的很窄的(5 个时间步长)窗口之中。图 13-7b 对图 13-7a 的复制神经网络提供时延解释，因此称为“时延神经网络”。642 这个网络共有 544 个突触权值。Lang and Hinton(1988)使用 TDNN 对四个孤立的词：

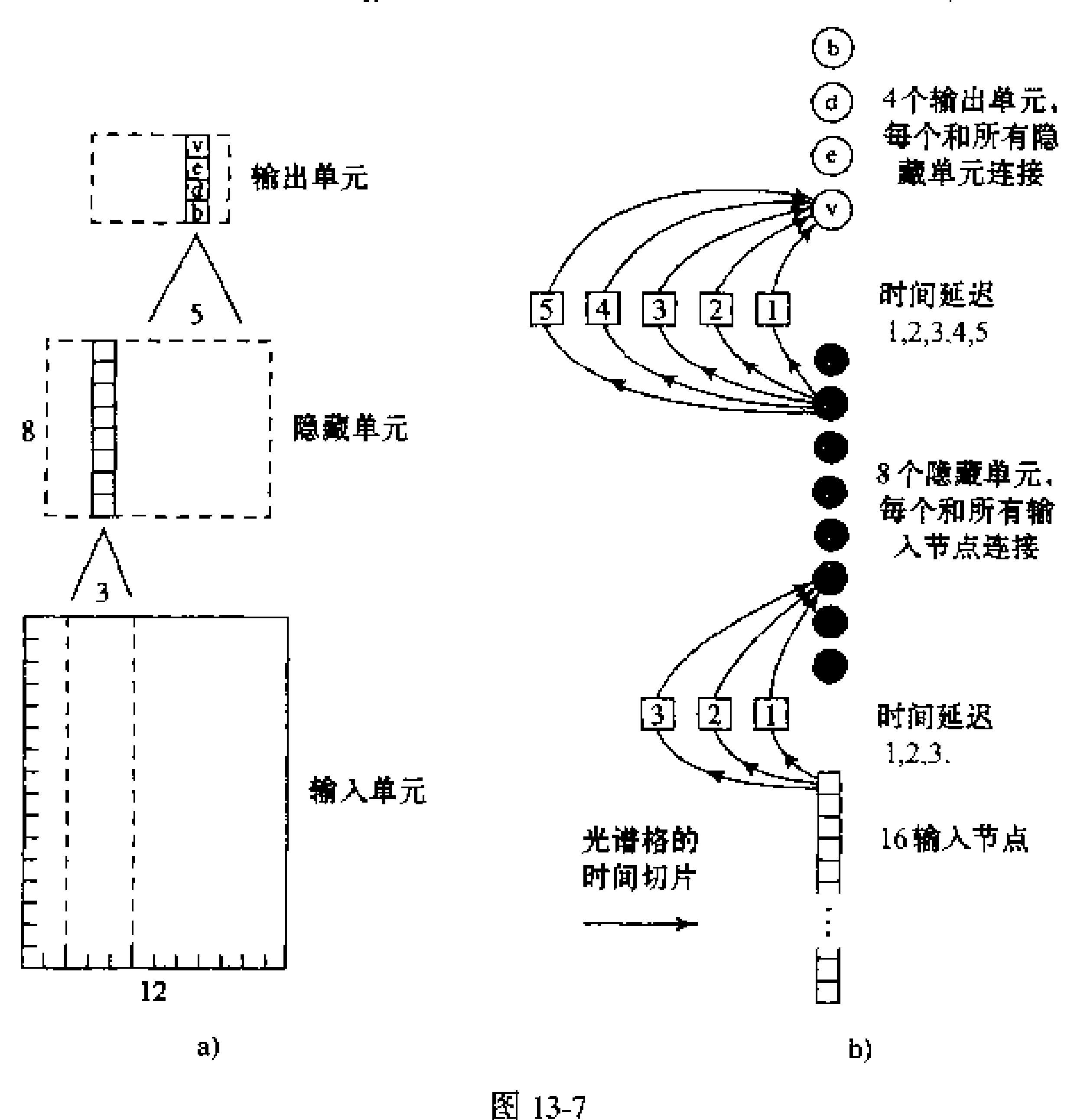


图 13-7

a)一个隐藏神经元和输出神经元沿时间复制的网络

b)时延神经网络(TDNN)表示(经允许，摘自 K.J.Lang and G.E.Hinton, 1988)

"bee","dee","ee","vee"进行识别，这要求在图 13-7 中使用四个输出神经元。通过使用不同于训练数据的测试数据获得了 93% 的识别率。在一个更精心研究的报告中 Waibel et al.(1989)使用了两个隐藏层，用于识别三个孤立的单词，"bee","dee"和"gee"。用三个人的发音作为测试集，在性能测试中，取得了平均 98.5% 的识别率。

TDNN 对于包含一串固定维数特征向量(比如音素)的时序模式的识别具有最好的效果。但是，在实际的语音识别器中，假设讲话的信号能被正确地切分为它的组成音素是不切合实际的。相反，对语音模式的超切分(super-segmented)时序结构恰当地建立模型是重要的。特别，语音识别器不得不去处理对于持续时间变化很大的词和句子片段以及非线性时序结构。要对语音信号的这些自然特征进行建模，语音识别领域的传统方法是使用一个状态转换结构，就像隐式 Markov 模型一样(Rabiner 1989;Jelinek,1997)。基本上，隐式 Markov 模型(hidden Markov model, HMM)表示由固有马尔可夫链产生的随机进程，以及与隐含状态相联系的一组观察分布；参见第 11 章注释[11]。在文献中已有很多混合型 TDNN 和 HMM 被研究[4]。

13.4 集中式时滞前馈网络

静态神经网络(如多层感知器，径向基函数网络)的原型应用是*结构化模式识别*。相反，*时序模式识别*要求对随时间演化的模式进行处理，对特定时刻的响应不仅依赖于输入的当前值，还依赖于以前的值。图 13-8 显示建立在静态神经网络上的*非线性滤波器*的框图(Mozer, 1994)。网络是通过短期记忆来模拟的。特别地，例如给定由输入信号的当前值 $x(n)$以及它的前 p 个值$x(n-1),\cdots,x(n-p)$组成的输入，它们存储在 p 阶延迟线记忆上，调整神经网络的自由参数使得网络输出 $y(n)$与期望响应 $d(n)$的平方误差达到最小。

图 13-8 所示的结构可以在单个神经元级或者一个神经元网络级来实现。这两情况分别在图 13-9 和图 13-10 给出。为了简化表达，我们用了抽头延迟线记忆作为图 13-9 和图 13-10 中的短期记忆结构。很明显，这两个图都可以通过使用传递函数 $G(z)$单元代替 z^{-1}来进行推广。 643

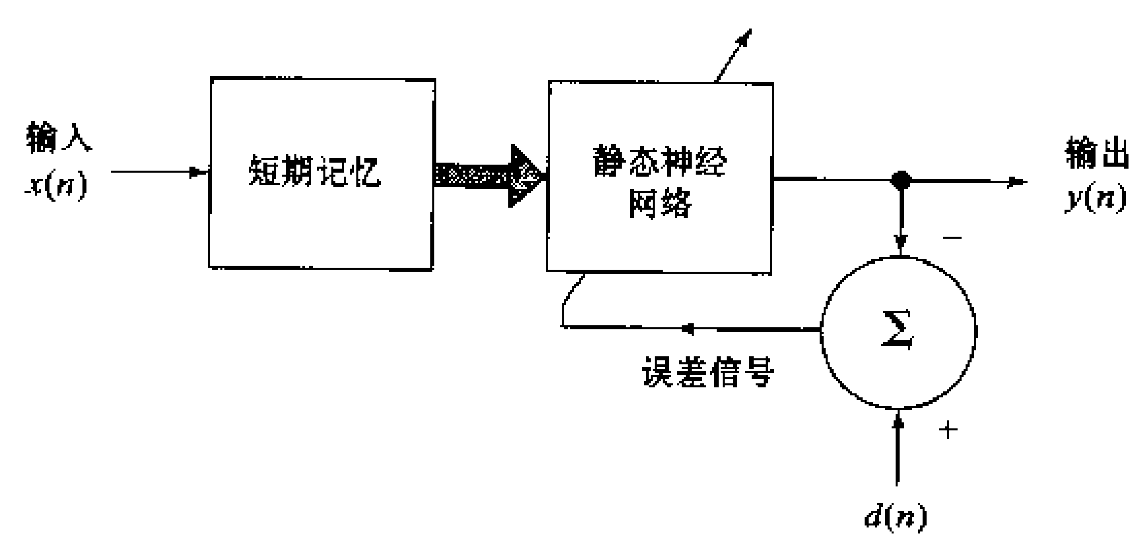


图 13-8 建立在静态神经网络上的非线性滤波器

图 13-9 中的时序处理单元是由其自己的抽头连接到神经元突触的抽头延迟线记忆组成的。抽头延迟线记忆捕获包含在输入信号中的时序信息并且神经元将那个信息嵌入到它们自己的突触权值中。图 13-9 中的处理单元称为*集中式神经滤波器*(focused neuronal filter)，集中的意义在于整个记忆结构都位于单元输入的末端。滤波器的输出，对输入 $x(n)$及其前面的值$x(n-1),\cdots,x(n-p)$的响应，由

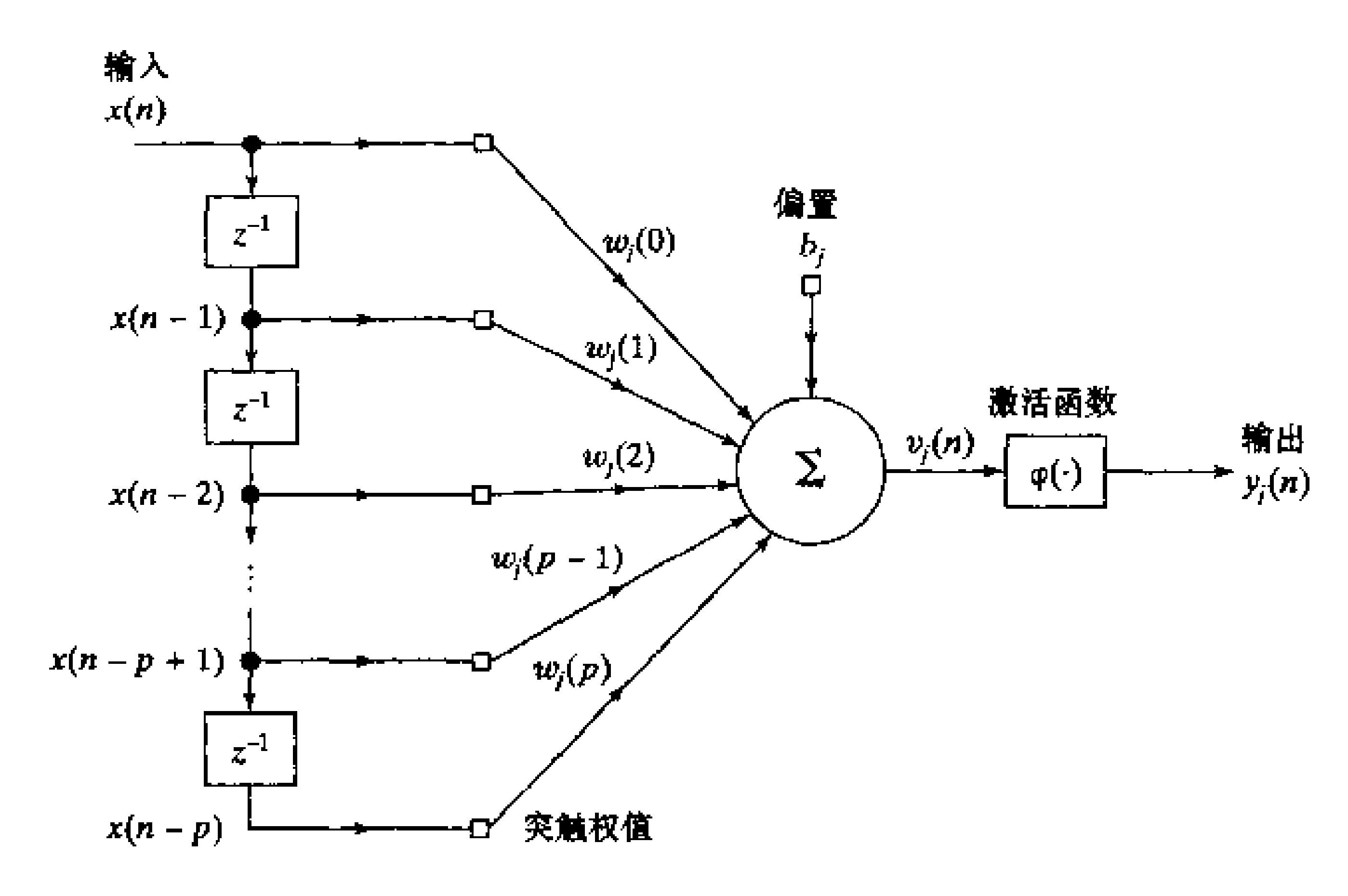


图 13-9 集中式神经元滤波器

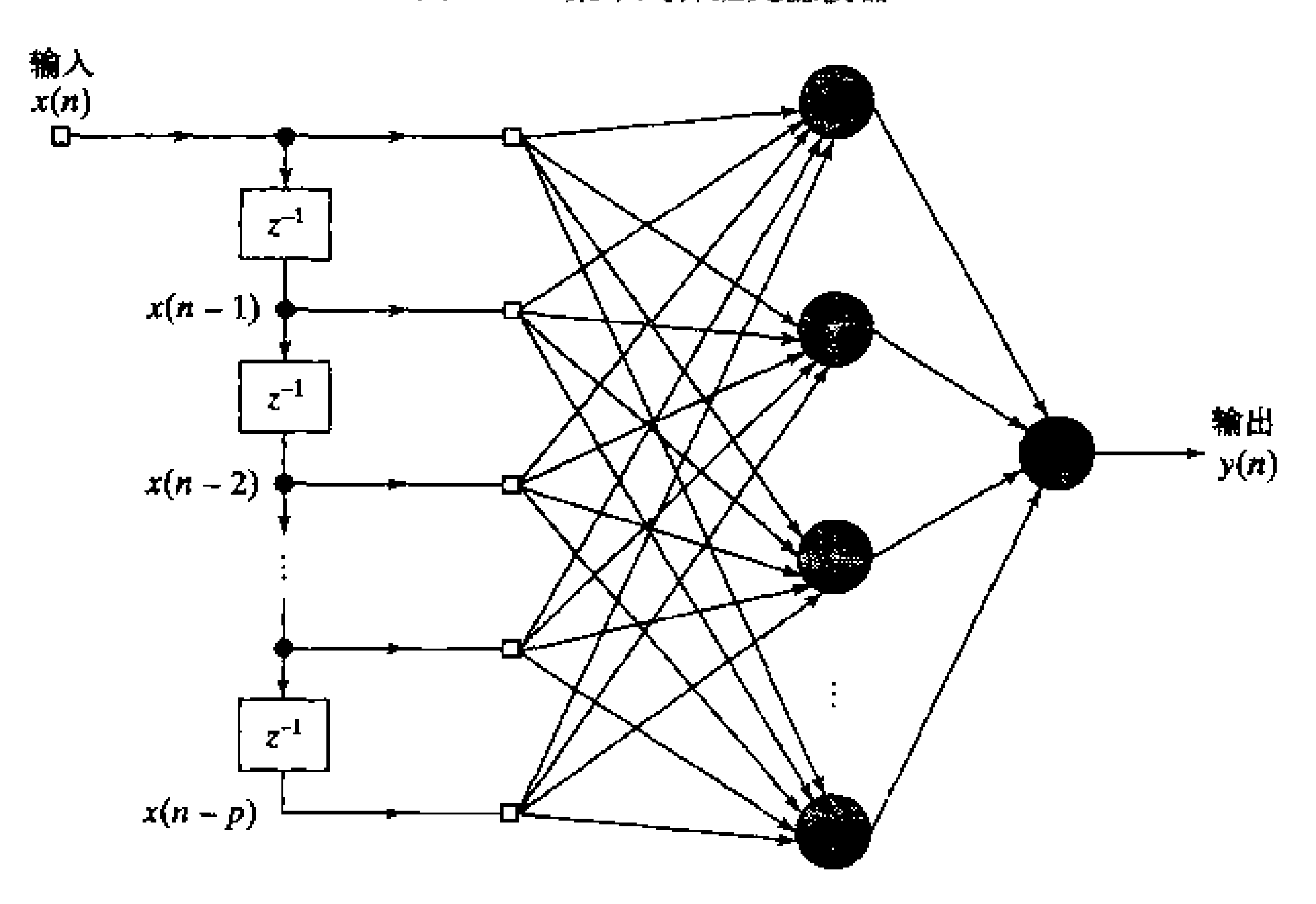


图 13-10 集中式时滞前馈神经网络(TLFN)；为表示方便省略了偏置层

$$y_j(n)=\varphi\Big(\sum_{l=0}^{p}w_j(l)x(n-l)+b_j\Big) \qquad (13.11)$$

给定，其中，$\varphi(\cdot)$是神经元 j 的激活函数，$w_j(l)$是突触权值，b_j 是偏置。注意激活函数的输入包含偏置加上输入样本和神经元的突触权值的卷积。

再看图 13-10，它是集中式时滞前馈网络(focused time lagged feedforward network, TLFN)，这里我们有一个更强大的非线性滤波器，包含有 p 阶的抽头延迟线记忆和多层感知器。要训练这个滤波器，我们可以使用第 4 章描述的标准的反向传播算法。在时刻 n，应用于网络输入层的"时序模式"即为信号向量

644

$$\mathbf{x}(n)=[x(n),x(n-1),\cdots,x(n-p)]^T$$

这可以看作对非线性滤波器在时刻 n 的状态描述。一个时段包括一系列状态(模式)，其数量由记忆阶 p 及训练样本的数量 N 决定。

如图 13-10 所示，假设多层感知器有一个隐藏层，非线性滤波器的输出由

$$y(n)=\sum_{j=1}^{m_1} w_j y_j(n)=\sum_{j=1}^{m_1} w_j \varphi\Big(\sum_{l=0}^{p} w_j(l)x(n-l)+b_j\Big)+b_o \qquad (13.12)$$

给出，其中集中式 TLFN 的输出神经元假定是线性的；输出神经元的突触权值由集合 $\{w_j\}_{j=1}^{m_1}$ 表示，m_1 是隐藏层的大小，b_o 为网络的偏置。

13.5 计算机实验

在这个计算机实验里，我们对图 13-10 中 TLFN 的使用进行研究，模拟一个困难的频率调制信号的时间序列：

$$x(n)=\sin(n+\sin(n^2)),n=0,1,2,\cdots$$

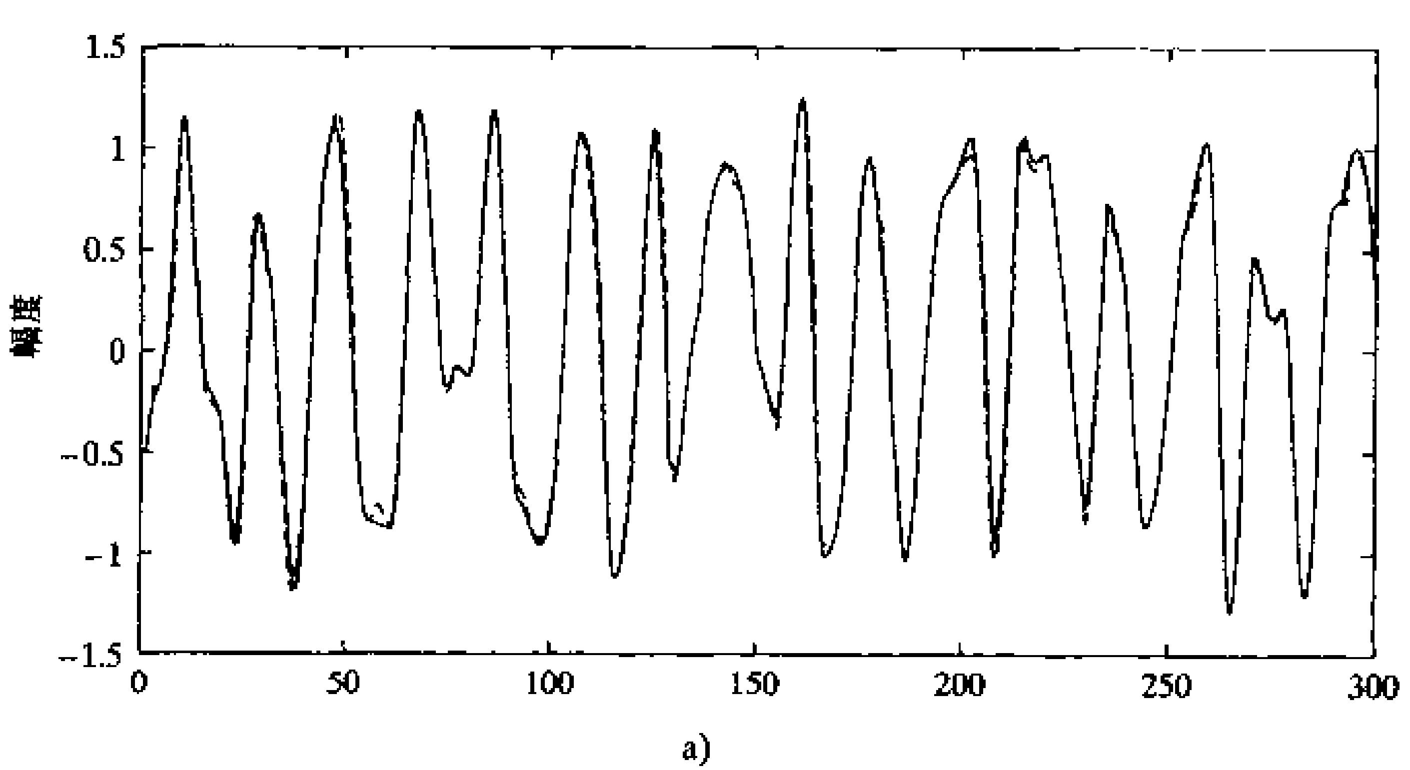
网络用作单步预测器，对于由集合 $\{x(n-l)\}_{l=0}^{p}$ 构成的输入，$x(n+1)$ 代表期望的响应。网络的组成及其参数如下：

抽头延时线记忆的阶 p：	20
隐藏层 m_1：	10 个神经元
隐藏层神经元的激活函数：	logistic 函数
输出层：	1 个神经元
输出神经元的激活函数：	线性函数
学习率参数(两层)：	0.01
动量常数：	无

645

用于训练网络的数据集有 500 个随机模式，每个模式含有从时间序列 $\{x(n)\}$ 中选择出来的 20 个时序样本。

图 13-11a 显示由网络对测试数据(以前未见过)执行的单步预测结果及实际波形的叠加。图 13-11b 显示预测的误差波形，这个误差定义为实际波形和预测波形之间的差别。预测误差的均方值为 1.2×10^{-3}。



a)

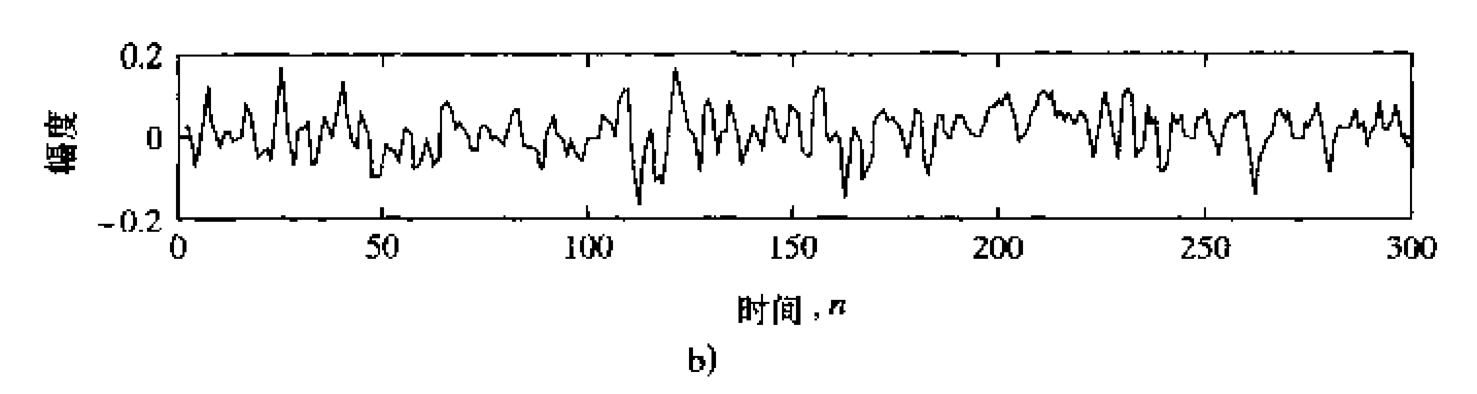


图 13-11 单步预测计算机实验结果

a)实际(实线)波形和预测(虚线)波形的叠加 b)预测误差的波形

13.6 通用短视映射定理

图 13-9 中的非线性滤波器可以推广为图 13-12 所示的滤波器。这个一般的动态结构包含两个功能模块。标号为$\{h_j\}_{j=1}^{L}$的模块表示时域的*多重卷积*，那就是说一个并行运行的线性滤波器组。h_j 是从一个较大的实值核集合中抽取出来的，每一个都代表着一个线性滤波器的脉冲响应。块标号为$\mathcal{N}$的模块表示静态的(即无记忆的)非线性前馈网络，如一个普通的多层感知器。图 13-12 中的结构是一个*通用动态映射器*(universal dynamic mapper)。在 Sandberg and Xu(1997a)中证明对于任何平移不变的*短视映射*(myopic map)，在适度的条件下利用图 13-12 描绘的结构能够以任意精度一致逼近。要求一个映射为短视的等价于"一致衰减记忆"；这里假设映射是*因果的*(causal)，这意味着一个只有在 $n=0$ 时应用输入信号时，才在时刻 $n\geq 0$ 由映射产生输出信号。对"平移不变"，我们是指如果 $y(n)$是映射对输入 $x(n)$产生的输出，那么对于平移输入 $x(n-n_0)$，映射的输出就是 $y(n-n_0)$，这里时间位移 n_0 是一个整数。在 Sandberg and Xu(1997b)中，他们进一步证明对单变量的、平移不变的、因果的和一致衰减的记忆映射，存在一个 Gamma 记忆和静态神经网络，它们的组合能够以任意精度一致逼近该映射。

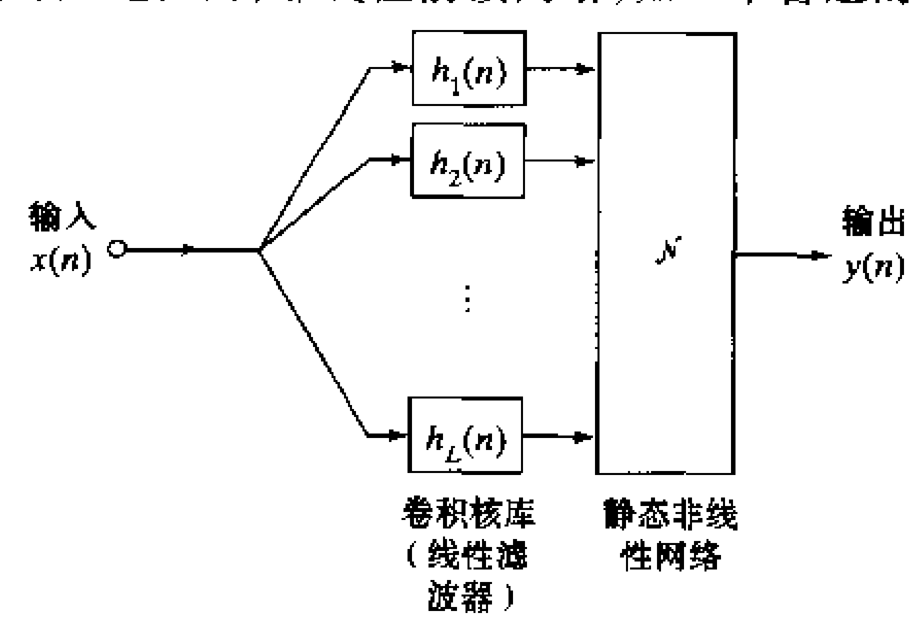


图 13-12 通用短视映射定理的一般结构

[646]

我们现在可以正式地将*通用短视映射定理*[15] 描述如下：

任何平移不变的短视动态映射可以由含有两个功能块的结构任意地一致逼近：一组线性滤波器馈给一个静态神经网络。

这个定理包含的结构可以采用集中式 TLFN 的形式。注意当输入和输出信号是有限数目变量的函数时(如图像处理)，定理依然成立。

通用短视定理有着很深的实际意义。它不仅对 NETtalk 及可能的 Gamma 记忆扩展提供数学基础，而且对更复杂的动态非线性处理模型的设计建立框架。在图 13-12 结构前端的多个卷积可以使用线性滤波器(通过有限冲激响应(FIR)或者无限冲激响应(IIR))来实现。对于静态神经网络，它可以用多层感知器、径向基函数网络或者支持向量机由第 4、5 和 6 章介绍

[647]

的训练算法来实现。换句话说，在那几章中给出的关于监督学习的资料基础之上，我们可以很自然地建立非线性滤波器或非线性动态过程的模型。最重要的是，假设线性滤波器本身是稳定的，图 13-12 中的结构是固有稳定的。因此，对于怎样处理短期记忆和无记忆非线性性，我们对它们的作用有清晰的分工。

13.7 神经元的时空模型

如图 13-9 所示的集中式神经滤波器在这里有一个很有意思的解释。单元延迟元素与相应的突触权值之间的组合可以看作是 p 阶的有限冲激响应(FIR)滤波器，如图 13-13a 所示。FIR 滤波器在数字信号处理中为一个基本的构件(Oppenheim and Schafer, 1989; Haykin and Van Veen, 1998)。相应地，图 13-9 中的集中式滤波器实际上是一个如图 13-13b 所示的非线性滤波器。如图 13-14 所示，在此表示基础上通过使用数量为 m_0 的多个输入我们可以扩充神经元的空间处理能力。图 13-14 是多输入神经元滤波器的时空模型。

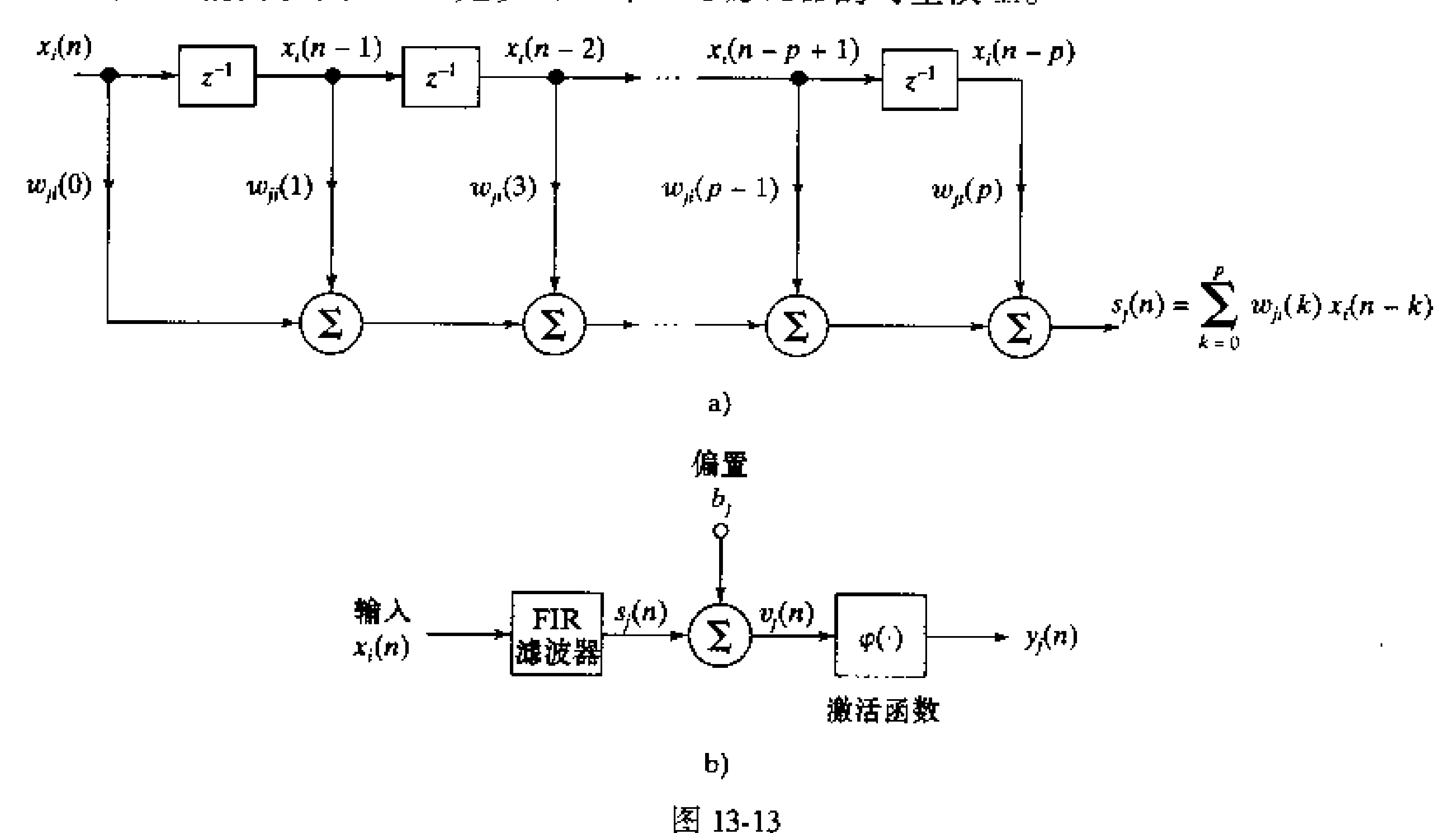


图 13-13

a)有限冲击响应(FIR)滤波器　b)神经元滤波器的非线性 FIR 滤波器解释

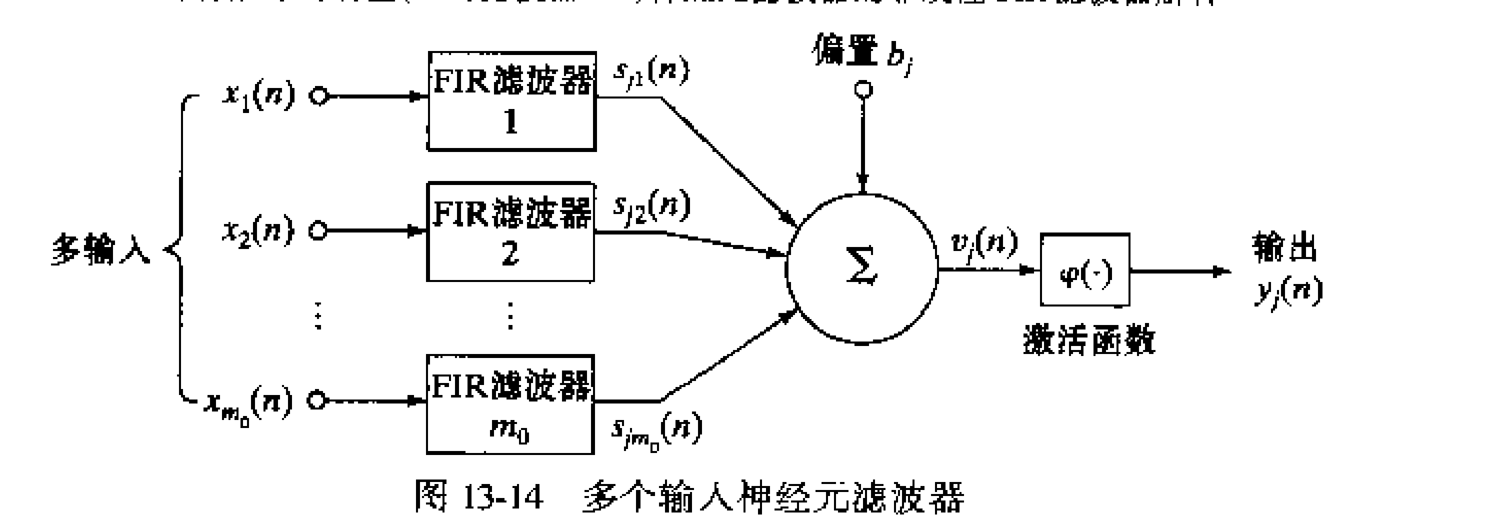


图 13-14　多个输入神经元滤波器

648

然而另一种描述图 13-14 的模型的方式是将其看作一个分布式神经元滤波器，这是在过滤行动在空间的不同点上是分布的意义之下。模型的时空特征描述如下：

- 神经元有 m_0 个"主"突触，每个主突触包含一个以 p 阶 FIR 形式实现的线性离散时

间滤波器；主突触计算信号处理的空间维。

- 每个主突触有$(p+1)$个“辅助”突触与各自的输入和FIR滤波器的记忆抽头相连接，它们计算信号处理的时间维。

在图13-14中的这个神经元滤波器的突触结构是树形的，如图13-15描述。整个突触权值的数目为$m_0(p+1)$。

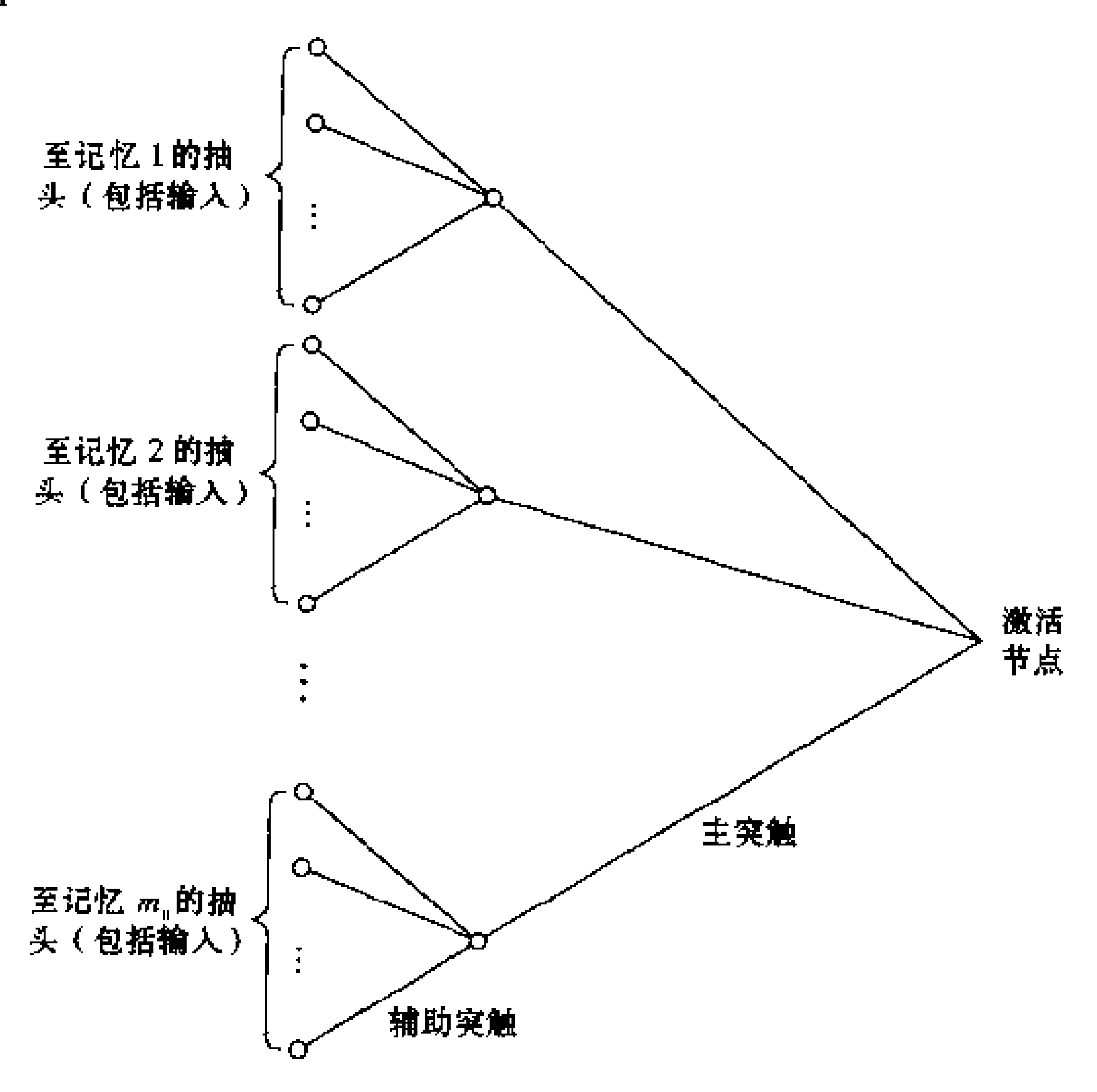


图13-15 多个输入神经元滤波器突触结构的树形描述

在数学术语中，我们可以将神经元滤波器执行的时空处理表达为

$$y_j(n)=\varphi\Big(\sum_{i=1}^{m_0}\sum_{l=0}^{p}w_{ji}(l)x_i(n-l)+b_j\Big) \tag{13.13}$$

其中$w_{ji}(l)$是属于第i个主突触的第l个辅助突触的权值，$x_i(n)$是在时刻n应用于第i个主突触的输入值，b_j为应用于该神经元的偏置。神经元的诱导局部域为$v_j(n)$，也就是在式(13.13)中的激活函数$\varphi(\cdot)$的变量，它可以看作对如下连续时间公式的离散时间“近似”：

649

$$v_j(t)=\sum_{i=1}^{m_0}\int_{-\infty}^{t}h_{ji}(\lambda)x_i(t-\lambda)d\lambda+b_j \tag{13.14}$$

在式(13.14)中积分是连续时间输入信号$x(t)$和表示突触i的线性连续时间滤波器的冲击响应$h_{ij}(t)$的卷积。式(13.14)是一个神经元诱导局部域时空行为的最通常描述方法。

加性模型

式(13.14)给出另外一种常用的神经元时空模型的基础。特别地，通过使用换算参数决定一个“典型的”突触冲击响应的符号和强度，我们简化神经元的时空模型，在此情况下有

$$h_{ji}(t)=w_{ji}\cdot h_j(t)\qquad 对所有的\ i \tag{13.15}$$

其中 $h_j(t)$将一个典型的后突触电位的时间特征模型化，并且 w_{ji}是一个标量，用于确定神经元 j 和输入 i 之间连接的符号(兴奋的或抑制的)和总强度(Shamma,1989)。这样在式(13.14)中代入式(13.15)，并且通过交换积分与求和次序，我们得到

$$v_j(t)=\int_{-\infty}^{t}h_j(\lambda)\Big(\sum_{i=1}^{m_0}w_{ji}x_i(t-\lambda)\Big)d\lambda+b_j=h_j(t)*\Big(\sum_{i=1}^{m_0}w_{ji}x_i(t)\Big)+b_j \tag{13.16}$$

其中星号 * 定义卷积。通用冲击响应 $h_j(t)$的形式依赖于要求的细节数量。一个常见选择为指数函数，定义为

$$h_j(t)=\frac{1}{\tau_j}\exp\Big(-\frac{t}{\tau_j}\Big) \tag{13.17}$$

其中 τ_j 是一个时间常量，它是神经元 j 的一个特征参数。式(13.17)中的时间函数 $h_j(t)$被看作是简单电路的冲击响应，该电路由电阻 R_j 和电容 C_j 组成，从一个电源得到馈给；即

$$\tau_j=R_jC_j \tag{13.18}$$

650

因此，我们使用式(13.16)和式(13.17)构造图 13.16 中模型的公式。使用物理术语，突触权值 $w_{j1},w_{j2},\cdots,w_{jm_0}$为电导率(即电阻的倒数)，而各自的输入 $x_1(t),x_2(t),\cdots,x_{m_0}(t)$则由电位(即电压)表示。求和连接由低输入电阻、单位电流增益和高输出电阻来表征；即它就是作为对输入电流进行求和的节点。因此馈入电阻 – 电容(Resistance-Capacitance, RC)电路的总电流为

$$\sum_{i=1}^{m_0}w_{ji}x_i(t)+I_j$$

其中第一个(求和)项是由于刺激 $x_1(t),x_2(t),\cdots,x_{m_0}(t)$分别作用于突触权值(电导率) $w_{j1},w_{j2},\cdots,w_{jm_0}$，而第二个项是表示外部作用偏置 b_j 的电源 I_j。

在神经网络文献中，图 13-16 中的神经元模型通常称为*加性模型*(additive model)。这个模型可以视为生物树突神经元的分布式传输线模型的块状电路近似(Rall,1989)。由于生物突触本身就是一个低通滤波器的良好近似，这也可以说明图 13-16 中的 RC 电路低通特性的合理性。

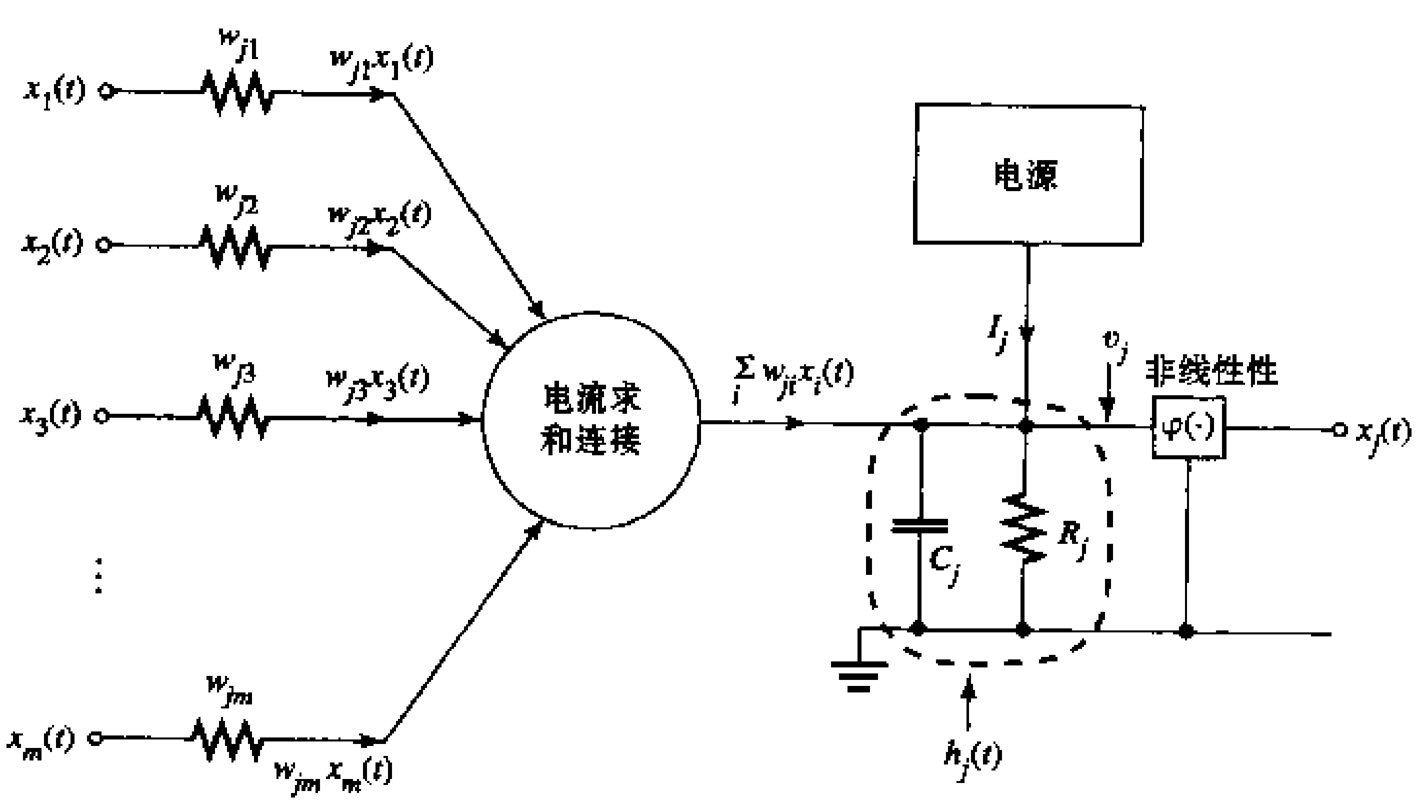


图 13-16 神经元的加性模型

13.8 分布式时滞前馈网络

通用短视映射算法提供集中式TLFN的数学基础，但它仅限于平移不变映射。这个局限性暗示集中式TLFN只适用于平稳(即时间不变)的环境。我们使用分布式时滞(distributed time lagged)前馈网络，可以克服这个局限，分布式的意义在于隐含的时间影响分布于整个网络。这样一个网络的结构基于图13-14的多个输入神经元滤波器作为神经元的时空模型。

令 $w_{ji}(l)$表示与FIR滤波器第 l 个抽头相连接的突触权值，该FIR滤波器模拟连接神经元 i 的输出到神经元j 的突触。下标 l 从0到 p，其中 p 是FIR的阶。依据这个模型，出现在第 j 个神经元的第 i 个突触输出的信号 $s_{ji}(n)$由卷积和

$$s_{ji}(n) = \sum_{l=0}^{p} w_{ji}(l)x_i(n-l) \tag{13.19}$$

给出，其中 n 表示离散时间。我们可以对于突触 i 分别引入下列状态向量和权值向量的定义，以矩阵的形式重写式(13.19)如下：

$$\mathbf{x}_i(n) = [x_i(n), x_i(n-1), \cdots, x_i(n-p)]^T \tag{13.20}$$

651

$$\mathbf{w}_{ji} = [w_{ji}(0), w_{ji}(1), \cdots, w_{ji}(p)]^T \tag{13.21}$$

这样我们可以把标量信号 $s_{ji}(n)$作为向量 $\mathbf{w}_{ji}(n)$和 $\mathbf{x}_i(n)$的内积，即

$$s_{ji}(n) = \mathbf{w}_{ji}^T\mathbf{x}_i(n) \tag{13.22}$$

对于输入向量 $\mathbf{x}_i(n)$， $i=1,2,\cdots,m_0$，式(13.22)定义图13-14模型中的神经元 j 的第 i 个突触的输出响应 $s_{ji}(n)$。向量 $\mathbf{x}_i(n)$被称为一种"状态"，因为它表示在时刻 n 第 i 个突触的条件。因此，对这个模型描绘的 m_0 个连接的全部贡献求和(即对下标 i 求和)，我们可以得到神经元 j 的输出 $y_j(n)$，表示为

$$v_j(n) = \sum_{i=1}^{m_0} s_{ji}(n) + b_j = \sum_{i=1}^{m_0} \mathbf{w}_{ji}^T\mathbf{x}_i(n) + b_j \tag{13.23}$$

$$y_j(n) = \varphi(v_j(n)) \tag{13.24}$$

其中 $v_j(n)$是神经元 j 的诱导局部域，b_j 是外部作用的偏置，$\varphi(\cdot)$是神经元的非线性激活函数。假设网络中所有的神经元都采用相同的非线性的形式。注意如果权值向量 $\mathbf{w}_{ji}$和状态向量 $\mathbf{x}_i(n)$分别由相应的标量 w_{ji}和x_i 代替，并且内积由普通的乘法运算代替，那么式(13.23)和式(13.24)中描述的动态模型就会化简为第4章中描述的普通多层感知器模型。

13.9 时序反向传播算法

为了训练分布式TLFN网络，我们需要一个监督学习算法，其中比较输出层每个神经元的每个时刻的实际响应与相应的期望(目标)响应。假设神经元 j 位于输出层，其实际响应是 $y_j(n)$，而这个神经元的期望响应为 $d_j(n)$，它们都在时刻 n 测量。我们可以定义该网络的平方误差和的瞬时值

$$\mathscr{E}(n) = \frac{1}{2}\sum_j e_j^2(n) \tag{13.25}$$

其中下标 j 仅指输出层的神经元，而 $e_j(n)$是误差信号，定义为

$$e_j(n) = d_j(n) - y_j(n) \tag{13.26}$$

对所有时间计算$\mathscr{E}(n)$的值然后求和，

$$\mathscr{E}_{\text{total}} = \sum_n \mathscr{E}(n) \tag{13.27}$$

652

目标是最小化这样定义的代价函数。为了计算最优权值向量估计值，记住达到这个目标的算法是基于最速下降方法的逼近。

处理这个问题的一个明显方法是将式(13.27)中的代价函数对 $\mathbf{w}_{ji}$ 进行微分，得到

$$\frac{\partial \mathscr{E}_{\text{total}}}{\partial \mathbf{w}_{ji}} = \sum_n \frac{\partial \mathscr{E}(n)}{\partial \mathbf{w}_{ji}} \tag{13.28}$$

为了利用瞬时梯度方法进一步处理，我们按时间展开网络。这里的策略首先是通过将其扩展成等价的但更大的"静态"网络，消除所有的延迟，接着应用标准反向传播算法计算瞬时误差梯度。不幸的是，这个方法受到下面几个负面性质所阻碍：

- 状态的前向传播与计算瞬时误差梯度所需项的反向传播之间失去对称意义。
- 传播误差项缺少一个好的递归公式。
- 需要全局纪录以跟踪哪些静态权值，它们实际上在展开分布式 TLFN 获得等价的网络中是相同的。

尽管用瞬时梯度估计是发展反向传播算法的时间形式的明显方法，从实用的观点来看这种方法不理想。

为克服上述瞬时梯度方法的问题，我们提下述处理(Wan,1990,1994)。首先，认识到把总误差梯度展开成如式(13.28)所示的瞬时误差梯度的和并不是惟一的。特别，可以考虑另一个表示代价函数$\mathscr{E}_{\text{total}}$对权值向量 $\mathbf{w}_{ji}(n)$的偏导数的方法，表示为

$$\frac{\partial \mathscr{E}_{\text{total}}}{\partial \mathbf{w}_{ji}} = \sum_n \frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial \mathbf{w}_{ji}} \tag{13.29}$$

其中时间下标 n 仅作用于$v_j(n)$。我们可以将偏导数$\partial\mathscr{E}_{\text{total}}/\partial v_j(n)$解释为在时刻 n 由于神经元j 的诱导局部域v_j 的一个变化而引起的代价函数的一个变化。然而重要的是注意

$$\frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial \mathbf{w}_{ji}} \neq \frac{\partial \mathscr{E}(n)}{\partial \mathbf{w}_{ji}}$$

只有当对所有的 n 求和，式(13.28)和(13.29)中的等式才成立。

给定式(13.29)的展开，我们现在可以使用权值空间的梯度下降的思想。特别，假设使用由

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) - \eta \frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial \mathbf{w}_{ji}(n)} \tag{13.30}$$

表示的递归形式的更新抽头－权值向量 $\mathbf{w}_{ji}(n)$，其中 η 是学习率参数。从式(13.23) 的定义看，我们可发现任何神经元 j，其诱导局部域 $v_j(n)$对权值向量 $\mathbf{w}_{ji}(n)$的偏导数由

$$\frac{\partial v_j(n)}{\partial \mathbf{w}_{ji}(n)} = \mathbf{x}_i(n) \tag{13.31}$$

653

给定，其中 $\mathbf{x}_i(n)$是应用于神经元 j 突触 i 的输入向量。此外，可以定义神经元 j 的局部梯度为

$$\delta_j(n) = -\frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} \tag{13.32}$$

因此，我们可以用一个熟悉的形式来重写式(13.30)：

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n)\mathbf{x}_i(n) \tag{13.33}$$

正如第4章中的标准反向转播算法所描述的那样，局部梯度的显式形式依赖于神经元 j 位于输出层还是隐藏层。这两种情况分别在下面讨论。

情形1　神经元 j 位于输出层

对于输出层而言，我们有

$$\delta_j(n) = \frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} = -\frac{\partial \mathscr{E}(n)}{\partial v_j(n)} = e_j(n)\varphi'(v_j(n)) \tag{13.34}$$

其中 $e_j(n)$是神经元 j 在输出处被测量的信号误差，而 $\varphi'(\cdot)$是激活函数 $\varphi(\cdot)$对其变量的导数。

情形2　神经元 j 是隐藏层神经元

当神经元 j 位于隐藏层时，我们定义$\mathscr{A}$为由神经元 j 以前向方式馈给其输入的神经元集合。令 $v_r(n)$表示属于集合$\mathscr{A}$的神经元 r 的诱导局部域。我们可以写成

$$\delta_j(n) = -\frac{\partial \mathscr{E}_{\text{total}}}{\partial v_j(n)} = -\sum_{r\in\mathscr{A}}\sum_k \frac{\partial \mathscr{E}_{\text{total}}}{\partial v_r(k)}\frac{\partial v_r(k)}{\partial v_j(n)} \tag{13.35}$$

其中我们已经使用下标 k 来代替 n 的位置以示特别注意之处。在式(13.35)里(用下标 r 代替 j)使用式(13.32)中的定义，可以得到

[654]

$$\delta_j(n) = \sum_{r\in\mathscr{A}}\sum_n \delta_r(k)\frac{\partial v_r(k)}{\partial v_j(n)} = \sum_{r\in\mathscr{A}}\sum_n \delta_r(k)\frac{\partial v_r(k)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} \tag{13.36}$$

其中 $y_j(n)$是神经元 j 的输出。我们知道偏导数$\partial y_j(n)/\partial v_j(n)$等于 $\varphi'(v_j(n))$，包括位于集合$\mathscr{A}$之外的神经元 j 这一点亦成立。所以可以将这一项提到双重求和式的外面，重写式(13.36)为

$$\delta_j(n) = \varphi'(v_j(n))\sum_{r\in\mathscr{A}}\sum_k \delta_r(k)\frac{\partial v_r(k)}{\partial y_j(n)} \tag{13.37}$$

像以前定义的那样，$v_r(n)$表示由神经元 j 的输出馈给的神经元 r 的诱导局部域。因此，为使式(13.19)和(13.23)的含义适于目前的情形，可以将 $v_r(k)$表示成

$$v_r(k) = \sum_{j=0}^{m_0}\sum_{l=0}^{p} w_{rj}(l)y_j(n-l) \tag{13.38}$$

在式(13.38)中已经包括用于神经元 r 的偏置 b_r，相当于 $j=0$ 时的项，定义为

$$w_{r0}(l) = b_r \quad \text{和} \quad y_0(n-l) = 1 \quad \text{对所有的 } l \text{ 和 } n \tag{13.39}$$

指标 p 定义式(13.38)内部和的上限，它是神经元 r，以及当前讨论的层中的所有其他神经元的每个突触滤波器的阶。指标 m_0 定义在式(13.38)中外部和的上限，是属于神经元 r 的所有主突触的数目。认识关于 l 的卷积和是可交换的。我们可重写式(13.38)为等价的形式

$$v_r(k) = \sum_{j=0}^{m_0}\sum_{l=0}^{p} y_j(l)w_{rj}(n-l) \tag{13.40}$$

上式对 y_j 进行求导，得到

$$\frac{\partial v_r(k)}{\partial y_j(n)} = \begin{cases} w_{rj}(k-l), & n \leqslant k \leqslant n+p \\ 0, & \text{其他} \end{cases} \tag{13.41}$$

按照式(13.41)，式(13.37)中的偏导数$\partial v_r(k)/\partial y_j(n)$，对于 n 在范围 $n \leqslant k \leqslant n+p$ 之外的

值，其值是0。对隐藏层中的神经元 j 来说，在式(13.37)中使用式(13.41)，得到

$$\delta_j(n) = \varphi'(v_j(n)) \sum_{r \in \mathcal{A}} \sum_{k=n}^{n+p} \delta_r(k) w_{rj}(k-l) = \varphi'(v_j(n)) \sum_{r \in \mathcal{A}} \sum_{l=0}^{p} \delta_r(n+l) w_{rj}(n) \tag{13.42}$$

定义一个新的 $(p+1)\times 1$ 维向量

$$\mathbf{\Delta}_r(n) = [\delta_r(n), \delta_r(n+1), \cdots, \delta_r(n+p)]^T \tag{13.43}$$

早些时候我们定义了式(13.21)中的权值向量 $\mathbf{w}_{ji}$。通过使用矩阵记号可以把式(13.42)重写成紧凑形式

$$\delta_j(n) = \varphi'(v_j(n)) \sum_{r \in \mathcal{A}} \mathbf{\Delta}_r^T(n) \mathbf{w}_{rj} \tag{13.44}$$

655

其中 $\mathbf{\Delta}_r^T(n)\mathbf{w}_{rj}$ 是向量 $\mathbf{\Delta}_r(n)$ 和 $\mathbf{w}_{rj}$ 的内积，这两个向量都是 $(p+1)$ 维的。式(13.44)完成在隐藏层中对于神经元 j 的 $\delta_j(n)$ 的计算。

我们现在可以总结权值更新方程为下述时序反向传播(temporal back propagation)关系(Wan,1990,1994)：

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n)\mathbf{x}_i(n) \tag{13.45}$$

$$\delta_j(n) = \begin{cases} e_j(n)\varphi'(v_j(n)), & j \text{ 为输出层} \\ \varphi'(v_j(n)) \sum_{r \in \mathcal{A}} \mathbf{\Delta}_r^T(n)\mathbf{w}_{rj}, & j \text{ 为隐藏层} \end{cases} \tag{13.46}$$

它可以推广为任意数量的隐藏层单元。立即可以看出这些关系式表示标准的误差反向传播算法的向量推广。如果我们用输入向量 $\mathbf{x}_i(n)$、权值向量 $\mathbf{w}_{rj}$ 以及局部梯度向量 $\mathbf{\Delta}_r$ 的标量形式来取代它们，那就变成了如第4章导出的标准反向传播算法。

为了计算位于隐藏层的神经元 j 的 $\delta_j(n)$，根据式(13.44)，我们通过那些兴奋是从神经元 j 导出的突触滤波器从后一层反向传播各个 δ。这个反向传播机制如图13-17所示。局部梯度 $\delta_j(n)$ 不是简单的由加权和得来，而是通过各主突触反向滤波形成的。特别地，对新的输入集合和期望响应向量，前向滤波器递增一个时间步，反向滤波器也一样。

656

图13-17 通过分布式TLFN的局部梯度的反向传播

我们现在能够看出这里使用时序反向传播算法的一些实际的好处：

1. 状态的前向传播和误差项的反向传播之间保持对称性，因而某种意义上并行分布式处理是可以维持的。

2. 每个惟一的突触滤波器权值在计算 δ 时只用一次。在瞬时梯度方法中遇到的项不存在冗余。

在推导式(13.45)和式(13.46)描述的时序反向传播算法[6]时，假设突触滤波器权值对所有的梯度计算都是固定的。在实际适应过程中这明显不是一个合法的假设。相应地，时序反向传播算法和使用瞬时梯度方法所得的时序形式之间将产生性能上的差异。然而，这些差异只是一个次要的属性。对一个较小的学习率参数 η，这两种算法中的学习特征的差异在实际应用中是可以忽略的。

因果性约束

细心检查式(13.42)可以发现 $\delta_j(n)$ 的计算是非因果性的，因为它需要各个 δ 和 $\mathbf{w}$ 未来值的知识。为了使这个计算为因果性的，首先注意用于适应调整的精确时间参照是无关紧要的。并且，网络中使用的突触结构都是 FIR 滤波器。因此，因果性要求使用附加的缓冲来暂存网络的内部状态。这样接下来我们要求所有权值的改变都基于误差信号的当前值和过去值。由此可以立即设置输出层神经元 j 的误差为 $\delta_j(n)$，接着改变那一层的突触滤波器权值。对前一层(即从输出层反向的一个隐藏层)，因果性约束暗示这一层神经元 j 的局部梯度

$$\delta_j(n-p) = \varphi'(v_j(n-p)) \sum_{r \in \mathcal{A}} \boldsymbol{\Delta}_r^T(n-p)\mathbf{w}_{rj} \tag{13.47}$$

的计算仅仅依赖于向量 $\boldsymbol{\Delta}_r$ 的当前值和过去值；那就是，

$$\boldsymbol{\Delta}_r(n-p) = [\delta_r(n-p), \delta_r(n+1-p), \cdots, \delta_r(n)]^T \tag{13.48}$$

式(13.47)是由式(13.46) 从第二行中将 n 用 $n-p$ 代替得到的，其中 p 是每个突触 FIR 滤波器的阶。像以前指出的那样，状态 $\mathbf{x}_i(n-p)$ 必须存储起来使得我们可以计算 $\delta_j(n-p)\mathbf{x}_i(n-p)$ 的积，这是为了改变连接最后一个隐藏层的神经元 j 和它前一层的神经元 i 的权值向量。对一个含多个隐藏层的网络来说，通过将时间平移两倍那么长，可以对更前一层(即输出层前面的两个层)继续这里描述的操作。操作以这种方式继续直到包括网络的所有计算层。

657

我们可以提出时序反向传播算法的因果形式，如表 13-1 中的小结。

表 13-1 时序反向传播算法小结

1. 向前逐层传播输入信号。确定输出层神经元 j 的误差信号 $e_j(n)$，这从期望响应中减掉实际输出得到。同时记录网络中每个突触的状态向量。

2. 对输出层神经元 j 计算：

$$\delta_j(n) = e_j(n)\varphi'_j(n)$$

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n)\mathbf{x}_i(n)$$

其中 $\mathbf{x}_i(n)$ 是与输出层神经元 j 相连的隐藏层神经元的突触 i 的状态。

3. 对隐藏层中的神经元 j，计算

$$\delta_j(n-lp) = \varphi'(v_j(n-lp)) \sum_{r \in \mathcal{A}} \boldsymbol{\Delta}_r^T(n-lp)\mathbf{w}_{rj}$$

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n-lp)\mathbf{x}_i(n-lp)$$

其中 p 是每个突触 FIR 滤波器的阶，同时指标 l 标识所讨论的隐藏层。特别，对有多个隐藏层的网络来说，$l=1$ 与紧靠输出层的第一个隐藏层相对应，$l=2$ 与紧靠输出层的两个隐藏层相对应，依次类推。

尽管这个算法在审美观点上不如式(13.45)和(13.46)中描述的非因果形式好，两种算法形式的基本不同只是在指标上的一些改变罢了。

总而言之，我们可以得到下列结论：

- 各个 δ 是通过网络的各层连续反向传播的，并且不增加延迟。这种传播强迫 δ 的内部值随着时间平移。
- 为了时间平移正确，状态(即 $\mathbf{x}_i(n)$的值)向量被保存起来，用于形成修改权值所需要的恰当乘积项。换句话说，只有状态向量是需要增加存储延迟，而执行 delta 的反向传播是不需要延迟的。
- 各个 δ 的反向传播与状态的前向传播保持对称。
- 和瞬时梯度方法一样，计算的阶对网络突触权值的数目是线性的。

分布式 TLFN 比在 13.4 节讨论的集中式 TLFN 更复杂。此外，用来训练分布式 TLFN 的时序反向传播算法计算量比适宜于训练集中式 TLFN 的标准反向传播算法的计算量更大。在最后的分析中，使用这两种方法中的哪一个，取决于需要解决的时序处理任务的环境是平稳的还是非平稳的[7]。 658

13.10 小结和讨论

对时序处理的需求出现在包括以下方面的大量应用中：

- 时间序列的预测和建模(Box, Jenkins, 1976; Haykin, 1996)。
- 噪声消除，其中需要一个主传感器(提供包含噪音的期望信号)以及一个参照传感器(提供噪音信号的一个相关形式)来消除噪声的影响(Widrow and Stearns, 1985; Haykin, 1996)。
- 未知通信信道的自适应均衡(Proakis, 1989; Haykin, 1996)。
- 自适应控制(Narendra and Annaswamy, 1989)。
- 系统辨识(Ljung 1987)。

当研究的系统或者是其固有的物理机制满足线性条件时，我们已经有一些很完善的理论来解决这些问题；可以参考上面提到的书。然而，如果一个系统或者物理机制是非线性的，我们面临的问题将更加困难。在这些情况下，神经网络有潜力提供行得通的解，从而在它们的应用中产生了很大的差异。

在神经网络的环景下，我们对时序处理有两种选择方案：

- 时滞前馈网络。
- 递归网络。

下面两章将讨论递归网络。这一章我们描述两类时滞前馈网络(TLFN)：集中式和分布式 TLFN。在一个集中式 TLFN 中，短期记忆完全位于静态网络的前端，可直接进行设计。训练集中式 TLFN，假定用多层感知器来实现静态神经网络，则可以用标准的反向传播算法完成。由 Sanberg and Xu(1997a, 1997b)得到的通用短视映射定理，我们有一个存在定理，通过用两个功能块(即一组线性滤波器块和一个静态神经网络)的级联，提供逼近任意短视映射(即具有一致衰减记忆的因果映射)的数学基础。这样一个结构可以使用集中式 TLFN 来实现，于是也就提供了这个定理的物理实现。

另外一类 TLFN 是分布式 TLFN，依赖于使用神经元的时空模型，即一个多输入神经元滤 659

波器。这个模型使用有限冲击响应(FIR)滤波器作为突触滤波器。这样，多输入神经元滤波器凭借在单个神经元周围建立的时空信号处理能力提供一个强大功能块。为了训练它，我们可以使用第3章描述的最小二乘(least-mean-square, LMS)算法。然而，要训练一个分布式TLFN，我们需要一个复杂的学习算法，诸如13.9节中描述的时序反向传播算法。分布式TLFN的突出特征是时间的隐式表示分布于整个网络中，因此具有处理非平稳(即时变)环境的能力。相反，在集中式TLFN中，按定义，时间的隐式表达集中于网络的前端，这限制它实际应用于平稳(即时间不变)环境。

注释和参考文献

[1] 关于时间在神经处理中的作用的短文，参见Elman(1990)的标题为“发现时间中的结构”的经典论文。

[2] 在Hopfield(1995)中描述在神经处理中时间的显式表示的一种方法。特别，对进行整体振荡的活动模式使用动作电位的定时表示模拟信息，并引用这方面神经生物学的证据；动作电位(action potentials)在第1章描述。

[3] 关于短期记忆结构和它们在时序处理的作用，参看Mozer(1994)。

[4] 对用于语音识别的TDNN和HMM的混合方法的讨论，可参看Bourlard and Morgan (1994)，Katagiri and McDermott (1996)和Bengio(1996)。

一些TDNN-HMM的混合结合使用TDNN框架编码器(即映射“听觉特征检测器”到一个“音素码”)和HMM词/句的路径发现器(即映射“音素符号”为“词/句的类”)，其中编码器和路径发现器都是单开设计的。在一些高级的TDNN-HMM混合中使用整个系统的平方误差损失函数使得和词/句的误差计数相关的损失能够被最小化。这后一种格式的例子为在Haffner et al.(1991)和Haffner(1994)中描述的多状态TDNN。分开设计模块的简单混合经常导致设计的训练性能和测试性能的不匹配。在这方面多状态TDNN表现更好。

在根本意义上递归网络(在第15章讨论)比类似TDNN的“复制”网络对于语音信号的时序结构建模具有更大的能力。但是，由于考虑到语音信号的非平稳性和非线性性，即使是递归网络，它们自己对于精确的语音识别也许并不是足够的。

[5] 关于通用短视映射定理的由来，参看Sandberg(1991)。

[6] 关于时序反向传播算法的另一个图解推导，参看Wan and Beaufays(1996)。

[7] 在Wan(1994)中，利用时序反向传播算法对NH_3激光的具有混沌震动的非平稳时间序列进行预测。这个特殊的时间序列是1992年在美国Santa Fe研究所举行的时间序列竞赛的一部分。对这个时序处理任务，包括标准的递归和前馈神经网络以及许多传统的线性技术在内的各种各样的解中，Wan的解赢得了竞赛(Wan, 1994)。混沌在第14章讨论。

习题

660

集中式时滞前馈网络(TLFN)

13.1 对用于非线性动态过程建模的集中式TLFN的主要特性进行概括。

13.2 在图13-10中描绘的集中式TLFN使用抽头延迟线记忆来实现短期记忆。那么在集中式TLFN中使用Gamma记忆来实现短期记忆的优缺点是什么？

13.3 在第2章中，我们定性地描述实现非线性自适应滤波的动态方法。这个方法涉及到一种静态神经网络，其刺激是通过应用滑动窗口来馈给输入数据。这个窗口随着每个新的数据样本的到来而发生移动，窗口中的旧样本滑出，给新的数据样本以空间。试讨论一个集中式 TLFN 如何实现这种连续学习的形式。

神经元的时空模型

13.4 考虑一个神经元滤波器，其诱导局部域 $v_j(t)$ 由式(13.16)定义。假设这个等式的时间函数 $h_j(t)$ 由平移单元冲击 $h_j(t)=\delta(t-\tau_j)$ 来代替，其中 τ_j 是一个固定延迟。描述这种修改对神经元滤波器带来的变化。

13.5 使用 LMS 算法，对图 13-9 中的多输入神经元滤波器给出学习算法的公式。

时序反向传播

13.6 图 13-18 描述用高斯形式的时间窗口作为时序处理的方法(Bodenhausen and Waibel, 1991)。与神经元 j 的突触 i 相联系的时间窗口，记为 $\theta(n,\tau_{ji},\sigma_{ji})$，其中 τ_{ji} 和 σ_{ji} 分别表示时延和窗口的宽度，表示为

$$\theta(n,\tau_{ji},\sigma_{ji}) = \frac{1}{\sqrt{2\pi}\sigma_{ji}}\exp\left(-\frac{1}{2\sigma_{ji}^2}(n-\tau_{ji})^2\right)$$

神经元 j 的输出模型为

$$y_j(n) = \varphi\left(\sum_{i=0}^{m_0} w_{ji}u_i(n)\right)$$

其中 $u_i(n)$ 是输入 $x_i(n)$ 和时间窗口 $\theta(n,\tau_{ji},\sigma_{ji})$ 的卷积。属于神经元 j 的突触 i 的权值 w_{ji} 和时延 τ_{ij} 都使用监督方式学习。

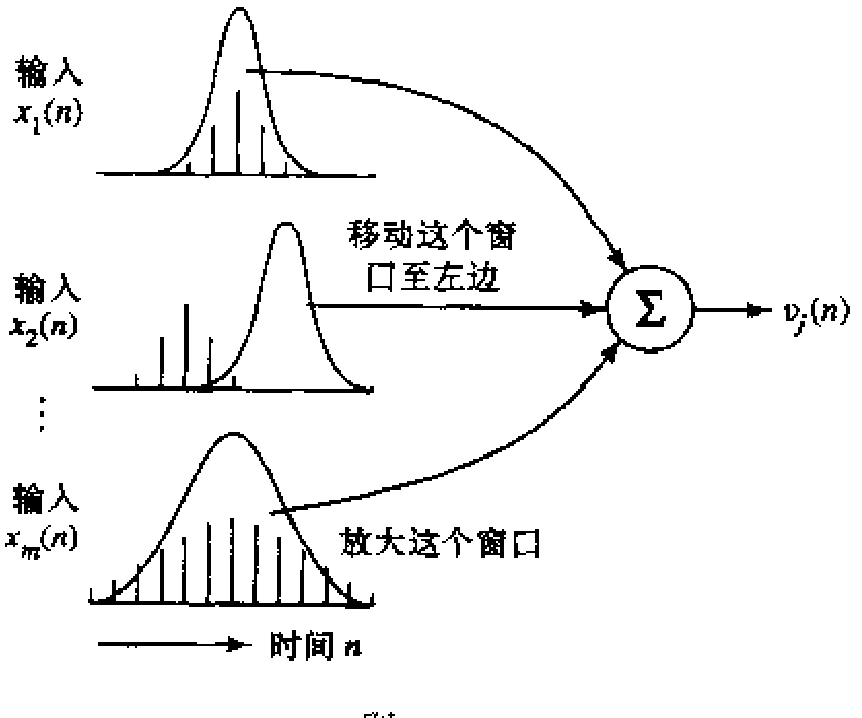


图 13-18

661

这个学习可以通过标准的反向传播算法来实现。试通过推导 $w_{ij},\tau_{ij},\sigma_{ij}$ 的更新公式演示这个学习过程。

13.7 在 13.9 节提供关于时序反向传播算法的材料中处理等长的突触 FIR 滤波器。你如何处理不等长的突触 FIR 滤波器?

13.8 讨论时序反向传播算法如何在单步预测的分布式 TLFN 的训练中使用。

13.9 约束的(因果的)和无约束的(非因果的)时序反向传播算法形式上的差异类似于标

准的最小二乘(LMS)算法和延迟LMS算法的差异；LMS算法在第3章讨论。扩充这个类比。

计算机实验

13.10 在这个题目里我们用标准的BP算法来解决困难的非线性预测问题，比较它与LMS算法的性能。时间序列由离散Volterra模型建立，其形式为

$$x(n)=\sum_i g_i v(n-i)+\sum_i\sum_j g_{ij} v(n-i)v(n-j)+\cdots$$

其中$g_i, g_{ij},\cdots$是Volterra系数。$v(n)$是独立的Gauss分布白噪声序列的抽样。$x(n)$是Volterra模型的输出。第一个求和项是我们熟悉的滑动平均(MA)时间序列模型，剩余的求和项是更高阶的非线性的部分。一般地，对Volterra系数的估计通常认为是困难的，主要是因为它们和数据的非线性关系。

在这个习题中，我们考虑一个简单的例子

$$x(n)=v(n)+\beta v(n-1)v(n-2)$$

时间序列是零均值的，不相关的，从而有一个白噪声的谱。然而，时间序列的样本并不是互相独立的。模型输出的方差由$\sigma_x^2=\sigma_v^2+\beta^2\sigma_v^4$给出，其中$\sigma_v^2$是白噪声的方差。

(a)构造一个多层感知器，有6个输入节点，隐藏层含有16个神经元，只有一个输出神经元。使用抽头延时线记忆馈给网络的输入层。隐藏层神经元使用sigmoid激活函数，限制在区间[0,1]区间内，而输出神经元充当一个线性的组合器。网络使用标准反向传播算法进行训练，有关参数如下：

662

学习率参数：	$\eta=0.001$
动量常数	$\alpha=0.6$
处理的样本总数	100 000
每个回合的样本数目	1 000
总的回合数目	2 500

白噪声方差σ_v^2为1。因此，用$\beta=0.5$，我们求出预测器的输出方差为$\sigma_x^2=1.25$。

计算非线性预测器的学习曲线，将预测器输出$x(n)$的方差绘制成训练样本的回合数的函数，一直画到2 500个回合。为了准备进行训练的每个回合，探讨下属两种方式：

(i)维持训练样本的时序，从一个回合到下一个回合与它产生的时序一样。

(ii)训练样本的顺序从一个状态(模式)到另一个状态是随机产生的。

同时，对1 000个样本的确认集使用交叉确认(在第4章中描述)，监测预测器的学习行为。

(b)重复试验，使用LMS算法对6个样本的输入执行线性预测。算法的学习率参数设置为$\eta=10^{-5}$。

(c)重复整个实验，用$\beta=1$，$\sigma_v^2=2$；接着再重复，用$\beta=2$，$\sigma_v^2=5$。

663

每个实验的结果应该揭示反向传播算法和LMS算法最初基本遵循相似的途径，然而反向传播算法继续改进，最终产生一个接近预定值σ_x^2的预测方差。

第 14 章 神经动力学

14.1 简介

在前一章关于时间处理中，我们研究了短时记忆结构和由记忆结构刺激静态神经网络(如多层感知机)，以及如何将它作为动态映射器运行。另一个可用于把时间以隐含的方式嵌入神经网络的运行之中的重要途径是通过使用反馈。把反馈应用于神经网络有两种基本途径：网络中单一神经元层次上的*局部反馈*，和包含整个网络的*全局反馈*。局部反馈处理起来是相对简单的，但全局反馈有更深的含义。在关于神经网络的文献中，带有一个或者更多反馈回路的神经网络被称为*递归网络*。在本章和下一章中，我们将注意力集中在使用全局反馈的递归网络。

反馈就像一柄双刃剑，因为如果你不能恰当地使用它，那么它就会产生负面效果。特别，反馈的应用能导致本来是稳定的系统变成不稳定的。在这一章中，我们的主要兴趣在于递归网络的稳定性。递归网络其他方面的问题我们将在下一章中考虑。

被视为非线性动力系统并特别强调稳定性问题的神经网络的主题被称为*神经动力学*(neurodynamics)(Hirsch,1989)。非线性动力系统的稳定性(或不稳定性)的一个重要特征就在于它是整个系统的特性。作为一个推论，*稳定性的存在总是意味着在系统的各个独立部分之间某种形式的协调*(Ashby,1960)。似乎对神经动力学的研究开始于 1938 年 Nicholas Rashevsky 的工作之中，那时将动力学应用于生物学领域第一次浮现在他充满幻想的头脑中。

非线性动态系统的稳定性是一个处理起来很棘手的问题。当谈到稳定性问题的时候，拥有工程背景的人经常会想到*有界输入和有界输出*(BIBO)的*稳定性准则*。依照这一准则，稳定性意味着如果有界的输入和初始条件或没有不必要干扰，那么系统的输出就必定不会无界地增长(Brogan,1985)。BIBO 稳定性准则非常适合于线性动态系统。但是，由于嵌入神经元结构之中的饱和非线性使得所有的这样一些非线性动态系统都是 BIBO 稳定的，所以把 BIBO 稳定性准则应用到神经网络上是无用的。

664

当在非线性动态系统背景谈到稳定性时，我们通常都意味着 *Lyapunov 意义的稳定性*。在 1892 年一个值得庆贺的日子里，Lyapunov(一位俄罗斯数学家和工程师)提出了众所周知的稳定性理论基本概念——*Lyapunov 直接方法*。这一方法被广泛用于线性和非线性系统中的稳定性分析，包括时不变和时变两种情况。因此，它可以直接用于神经网络中的稳定性分析。事实上，本章中提到的很多材料都涉及到 Lyapunov 直接方法。但是，它的应用不是一个轻松的任务。

对神经动力学的研究可能会遵从两种途径之一，这取决于实际的应用：

- *确定性神经动力学*：此时神经网络模型带有确定的行为。数学上用一组非线性微分方程来描述，微分方程定义作为时间函数的模型的精确进化(Grossberg,1967;Cohen and Grossberg,1983;Hopfield,1984)。
- *统计性神经动力学*：此时神经网络受到存在噪声的扰动。在这种情况下，我们将不

得不处理随机非线性微分方程组，因而用概率术语表示解(Amari et al.,1972;Peretto,1984;Amari,1990)。随机性和非线性的组合使得这个主题将非常难于处理。

在本章中，我们将自己限制在确定性神经动力学之内。

本章的组织

本章中的材料分成三个部分。在由14.2节到14.6节组成本章的第一部分，我们提供介绍性的材料。14.2节介绍一些动态系统中的基本概念，随后在14.3节中讨论的平衡点稳定性。14.4节中描绘在动态系统研究中浮现出的各种类型的吸引子。在14.5节再次讨论曾经在第13章中导出的神经元的加性模型。在14.6节讨论作为神经网络范例的吸引子的运作。

本章第二部分由14.7节到14.11节组成，处理联想记忆。14.7节致力于详细讨论Hopfield模型和作为按内容寻址记忆使用的离散Hopfield模型的细节问题。在14.8节提出Hopfield网络这种应用上的计算机实验。14.9节中对于包含Hopfield网络和其他联想记忆的非线性动态系统作为其特例的非线性系统，给出它们的Cohen-Grossberg定理。在14.10节中描述另一个被称为盒中脑状态模型的神经动力学模型，该模型非常适用于聚类。14.11节提出对这个第二种模型上的计算机实验。

665

最后部分由14.12节到14.14节组成，处理混沌的论题。14.12节讨论混沌过程的不变特征，随后在14.13节讨论混沌过程动力学重建这一紧密相关题目。动力学重建的计算机实验在14.14节中给出。

本章在14.15节中用一些最后评论结束本章。

14.2 动态系统

为了进行神经动力学的研究，我们需要用一个数学模型描述非线性系统的动力学。自然最适合这一用途的模型就是状态空间模型。根据这个模型，我们考虑一组状态变量，假设这些变量的值(在任意特定时刻)都包含充分的信息可以预测系统的可能演化。令$x_1(t), x_2(t), \cdots, x_N(t)$，表示非线性动态系统的状态变量，其中连续时间$t$是独立变量且$N$为系统的阶。为了简化符号，把这些状态变量收集在一个叫做系统状态向量的$N\times 1$的向量$\mathbf{x}(t)$里。那么非线性动态系统的一大类的动力学特性就可以用一阶微分方程组

$$\frac{d}{dt}x_j(t) = F_j(x_j(t)), \qquad j = 1,2,\cdots,N \tag{14.1}$$

的形式给出，一般来说，其中的函数$F_j(\cdot)$是它的自变量的非线性函数。我们可以用向量符号把这个方程组写成紧凑形式

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{F}(\mathbf{x}(t)) \tag{14.2}$$

其中非线性函数$\mathbf{F}$是向量值的，它的每一个元素作用于下述状态向量中的一个对应元素：

$$\mathbf{x}(t) = [x_1(t), x_2(t), \cdots, x_N(t)]^T \tag{14.3}$$

如在式(14.2)中那样，若向量函数$\mathbf{F}(\mathbf{x}(t))$不显式地依赖于时间$t$，则这样的非线性动态系统被称为自治的(autonomous)；否则称为非自治的(nonautonomous)[1]。我们只关注自治系统。

不管非线性函数 $\mathbf{F}(\cdot)$的精确形式是什么，状态向量 $\mathbf{x}(t)$必须随时间改变；否则，$\mathbf{x}(t)$就是常量而系统也不再是动态的。因此我们可以正式定义一个动态系统如下：

动态系统是状态随时间变化的系统。

此外，我们可以把 $d\mathbf{x}/dt$ 作为"速度"考虑，不是在物理意义上而是在抽象意义上的。那么，根据式(14.2)，可以将向量函数 $\mathbf{F}(\mathbf{x})$ 称为速度向量场或者简单地称为*向量场*(vector field)。

状态空间

将状态空间方程(14.2)看作描述 N 维状态空间中一个点的运动是有益的。状态空间可能是欧几里德空间或者是它的一个子集。也可能是非欧氏空间，就像圆、球、环或者其他一些微分流形。但是，我们的兴趣只限于欧氏空间。 666

状态空间很重要，因为它给我们提供可视的/概念化的工具用来分析由式(14.2)描述的非线性系统的动力学。它是通过把我们的注意力集中于运动的全局特性而不是方程的解析解或数值解的细节方面来实现的。

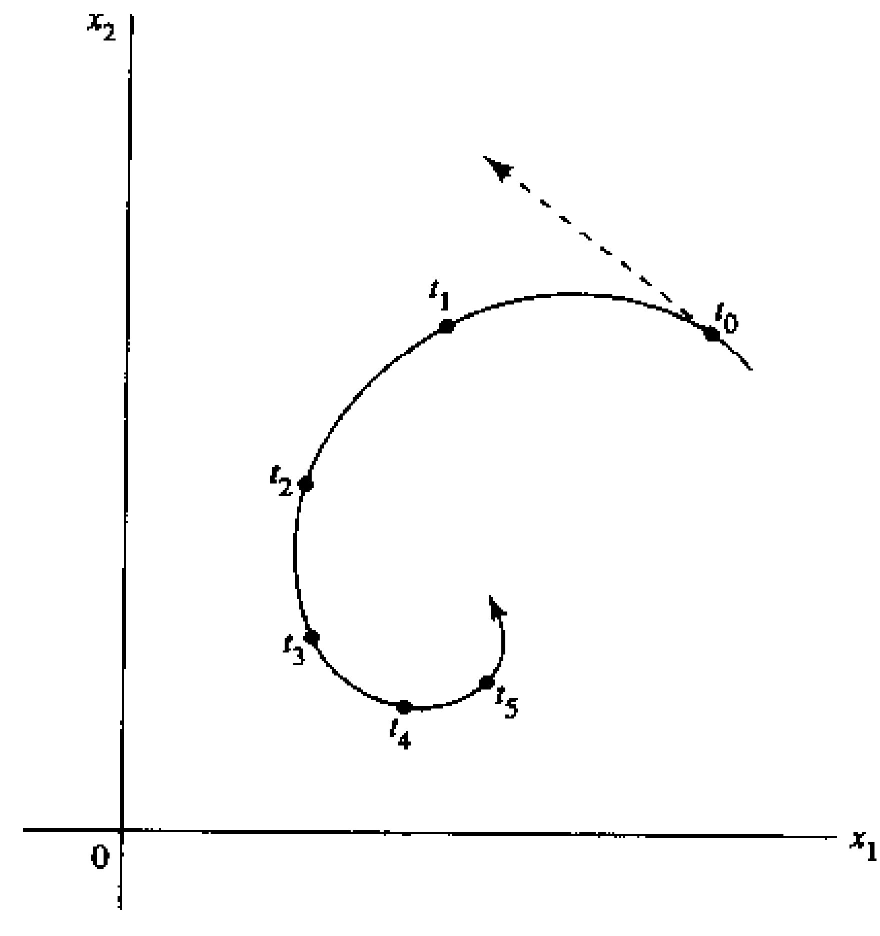


图 14-1 二维动态系统的轨线(轨道)

在一特定时刻 t，用 N 维状态空间中的一个点表示系统被观察状态(即状态向量 $\mathbf{x}(t)$)。用状态空间中的一条曲线表示系统状态随时间 t 的变化，曲线上的每一点都(显式地或隐含地)带有记录观察时间的标记。这条曲线叫做系统的*轨线*或*轨道*。图 14-1 描绘一个二维系统的轨线。轨线的瞬时速度(即速度向量 $d\mathbf{x}(t)/dt$)用切向量表示，如图 14-1 中 $t = t_0$ 时刻用虚线的表示。因此我们可以得出轨线上每一点的速度向量。

由不同初始条件产生的不同轨线的集合称为系统的*状态相图*(state portrait)。状态相图包含状态空间中所有那些定义向量场 $\mathbf{F}(\mathbf{x})$的点。注意对于自治系统来说，每种初始状态将只有一条轨线穿过。从状态相图产生的一个有用概念是动态系统的*流*(flow)，被定义为状态空间在系统内部的运动。换句话说，可以想像一下状态空间在自身内部流动，就像一种流体，每一个点(状态)沿着一条特定轨线的流动 (Abraham and Shaw, 1992)。这里描述的流的思想在图 14-2 的状态相图中有生动的说明。

给定一个动态系统的状态相图，可以构造一个对应于状态空间中每一个点的速度(切线)向量场。这样得到的图也提供系统中向量场的描绘。图 14-3 中显示许多速度向量，展现完全的场看起来像什么样子。向量场的用处在于事实上它通过在状态空间中每一个特定点以惯性速度移动，给我们提出一种对动态系统固有运动倾向的可视描述。 667

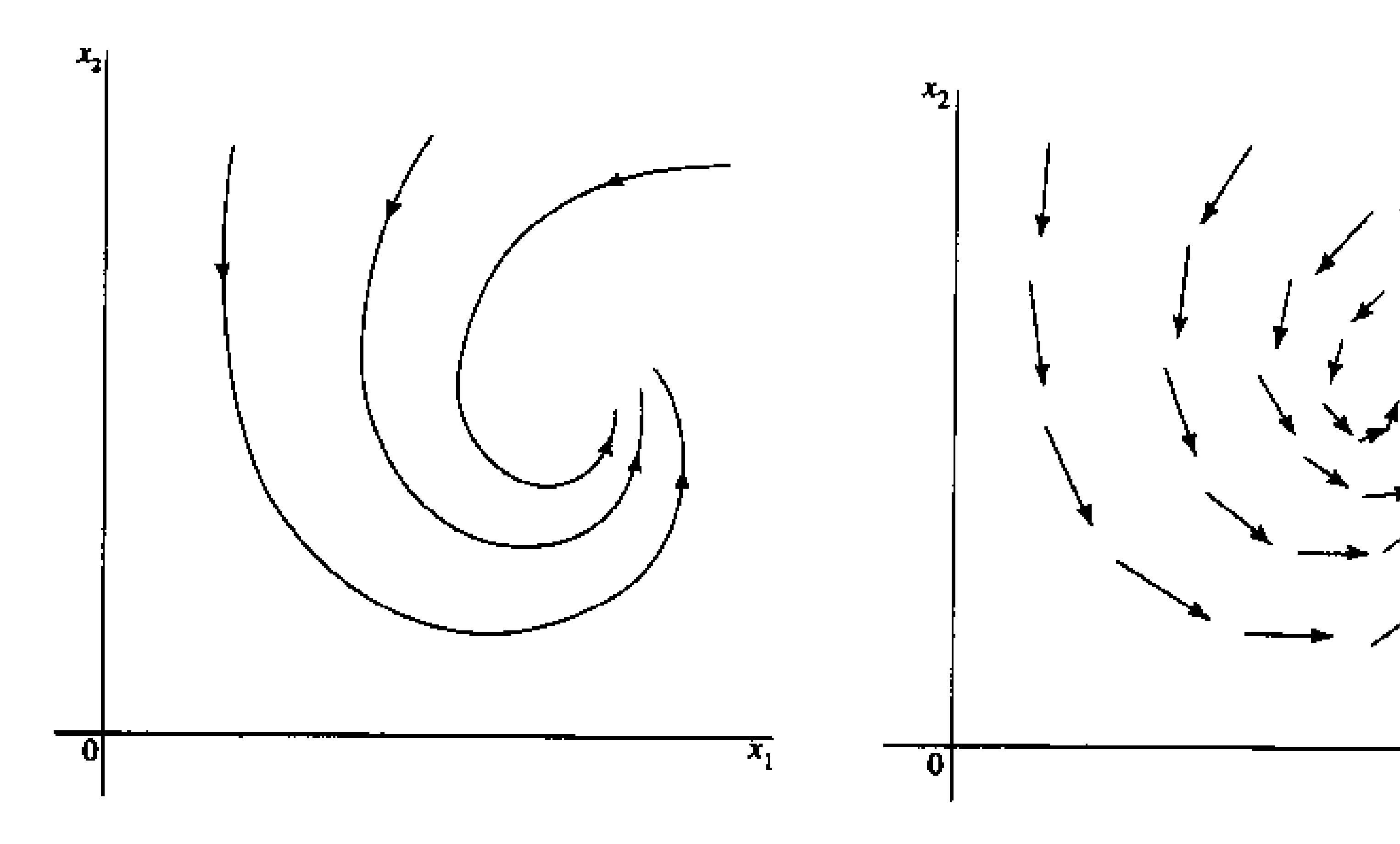


图14-2　二维动态系统的状态(相位)图　　　　图14-3　二维动力系统向量场

Lipschitz 条件

为了状态空间方程(14.2)有解且是惟一解，必须在向量函数 $\mathbf{F}(\mathbf{x})$上施加一定的限制。为了便于表示，我们已经舍弃了状态向量 $\mathbf{x}$ 对时间 t 的依赖，而这是我们一次又一次遵从的惯例。存在解的充分条件为 $\mathbf{F}(\mathbf{x})$对它的所有自变量是连续函数。然而，它这一限制本身不足以保证解的惟一性。为了做到这一点，我们必须施加被称为 Lipschitz 条件的额外限制。令 $\|\mathbf{x}\|$ 表示向量 $\mathbf{x}$ 的范数或者欧几里德长度。令 $\mathbf{x}$ 和 $\mathbf{u}$ 作为赋范向量(状态)空间上某一开集 $\mathcal{M}$上的一个向量对。然后，根据 Lipschitz 条件，存在一个常量 K 使得下式对$\mathcal{M}$中所有的 $\mathbf{x}$ 和 $\mathbf{u}$ 都成立(Hirsch and Smale,1974;E.A.Jackson,1989)：

$$\|\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{u})\| \leqslant K\|\mathbf{x}-\mathbf{u}\| \tag{14.4}$$

满足式(14.4)的向量值函数 $\mathbf{F}(\mathbf{x})$被称为满足 Lipschitz 条件，K 叫做 $\mathbf{F}(\mathbf{x})$的 Lipschitz 常数。式(14.4)也意味着函数 $\mathbf{F}(\mathbf{x})$关于 $\mathbf{x}$ 的连续性。因此，对自治系统来说，Lipschitz 条件是状态空间方程(14.2)存在且只存在惟一解的充分条件。特别地，如果所有偏导数$\partial F_i/\partial x_j$ 处处有限，则函数 $\mathbf{F}(\mathbf{x})$满足 Lipschitz 条件。

散度定理

考虑自治系统状态空间中某个体积 V 和曲面 S 的区域，并且设想由区域的点组成的"流"。从以前的讨论，我们认识到速度向量 $d\mathbf{x}/dt$ 和向量场 $\mathbf{F}(\mathbf{x})$是相等的。倘若体积 V 内的向量场 $\mathbf{F}(\mathbf{x})$是相当光滑，则可以从向量微积分学的角度应用散度定理(Jackson,1975)。令 $\mathbf{n}$ 表示曲面 S 上某小块 dS 处指向所包含体积外部的单位法向量。然后，根据散度定理，关系式

$$\int_S(\mathbf{F}(\mathbf{x})\cdot\mathbf{n})\,dS=\int_V(\nabla\cdot\mathbf{F}(\mathbf{x}))\,dV \tag{14.5}$$

在 $\mathbf{F}(\mathbf{x})$散度的体积分和 $\mathbf{F}(\mathbf{x})$向外法线分量的曲面积分之间成立。式(14.5)左端的值被认为

668

是从曲面 S 所包围的区域中流向外部的净流量。如果该值为零，则说系统是保守的(conservative)；若为负，则说系统是耗散的(dissipative)。根据式(14.5)，同样可以说，如果散度$\nabla \cdot \mathbf{F}(\mathbf{x})$(一个标量)为零则系统是保守的，若为负则系统是耗散的。

14.3 平衡状态的稳定性

考虑由状态空间方程(14.2)描述的自治动态系统。一个常向量$\bar{\mathbf{x}} \in \mathcal{M}$称为系统的平衡(稳定)状态，如果条件

$$\mathbf{F}(\bar{\mathbf{x}}) = \mathbf{0} \tag{14.6}$$

669

满足，其中的$\mathbf{0}$为零向量。速度向量 $d\mathbf{x}/dt$ 在平衡状态 $\bar{\mathbf{x}}$ 处消失，因此常量方程 $\mathbf{x}(t) = \bar{\mathbf{x}}$ 是方程(14.2)的解。此外，由于解的惟一性，没有其他的解曲线能够穿过平衡状态 $\bar{\mathbf{x}}$。平衡状态也称为奇异点，表示在平衡点这种情况下，轨线将会退化到这个点本身。

为了加深对平衡条件的理解，假设非线性函数 $\mathbf{F}(\mathbf{x})$对于状态空间方程(14.2)来说足够光滑，使得在 $\bar{\mathbf{x}}$ 的邻域可以作为线性函数处理。特别，令

$$\mathbf{x}(t) = \bar{\mathbf{x}} + \Delta\mathbf{x}(t) \tag{14.7}$$

其中的 $\Delta\mathbf{x}(t)$是 $\bar{\mathbf{x}}$ 的微小偏差。然后，保留 $\mathbf{F}(\mathbf{x})$的 Taylor 级数展开中的前两项，将其近似为

$$\mathbf{F}(\mathbf{x}) \simeq \bar{\mathbf{x}} + \mathbf{A}\Delta\mathbf{x}(t) \tag{14.8}$$

矩阵 $\mathbf{A}$ 是非线性方程 $\mathbf{F}(\mathbf{x})$的 Jacobi 矩阵，在 $\mathbf{x} = \bar{\mathbf{x}}$ 点处计值，表示为

$$\mathbf{A} = \frac{\partial}{\partial \mathbf{x}} \mathbf{F}(\mathbf{x}) \Big|_{\mathbf{x}=\bar{\mathbf{x}}} \tag{14.9}$$

将式(14.7)和式(14.8)代入式(14.2)，然后使用平衡状态的定义，我们得到

$$\frac{d}{dt}\Delta\mathbf{x}(t) \simeq \mathbf{A}\Delta\mathbf{x}(t) \tag{14.10}$$

670

倘若 Jacobi 矩阵 $\mathbf{A}$ 是非奇异的，即逆矩阵 $\mathbf{A}^{-1}$存在，式(14.10)描述的近似值足以确定系统轨线在平衡状态 $\bar{\mathbf{x}}$ 邻域的局部性质。如果 $\mathbf{A}$ 是非奇异的，则平衡状态的性质主要取决于 $\mathbf{A}$ 的特征值，因此可以根据它的相应方式进行分类。特别，当 Jacobi 矩阵 $\mathbf{A}$ 的特征值有 m 个带有正实数部分，我们可以说平衡状态 $\bar{\mathbf{x}}$ 属于类型(type)m。

对于二阶系统这种特殊情况而言，平衡状态的分类可归结为表 14-1 所列情况，相应相图表示在图 14-4 中(Cook，1986；Arrowsmith and Place，1990)。不失一般性，假设平衡状态位于状态空间的原点，也就是 $\bar{\mathbf{x}} = \mathbf{0}$ 的地方。注意对于图 14-4e 中的鞍点，通向鞍点的轨线是稳定的，而从鞍点离开的轨线则是不稳定的。

表 14-1 二阶系统平衡状态的分类

平衡状态 $\bar{\mathbf{x}}$ 的类型	Jacobi 矩阵 $\mathbf{A}$ 的特征值
稳定结点	负实数
稳定焦点	实部为负的共轭复数
不稳定结点	正实数
不稳定焦点	实部为正的共轭复数
鞍点	不同号的实数
中心	共轭纯虚数

671

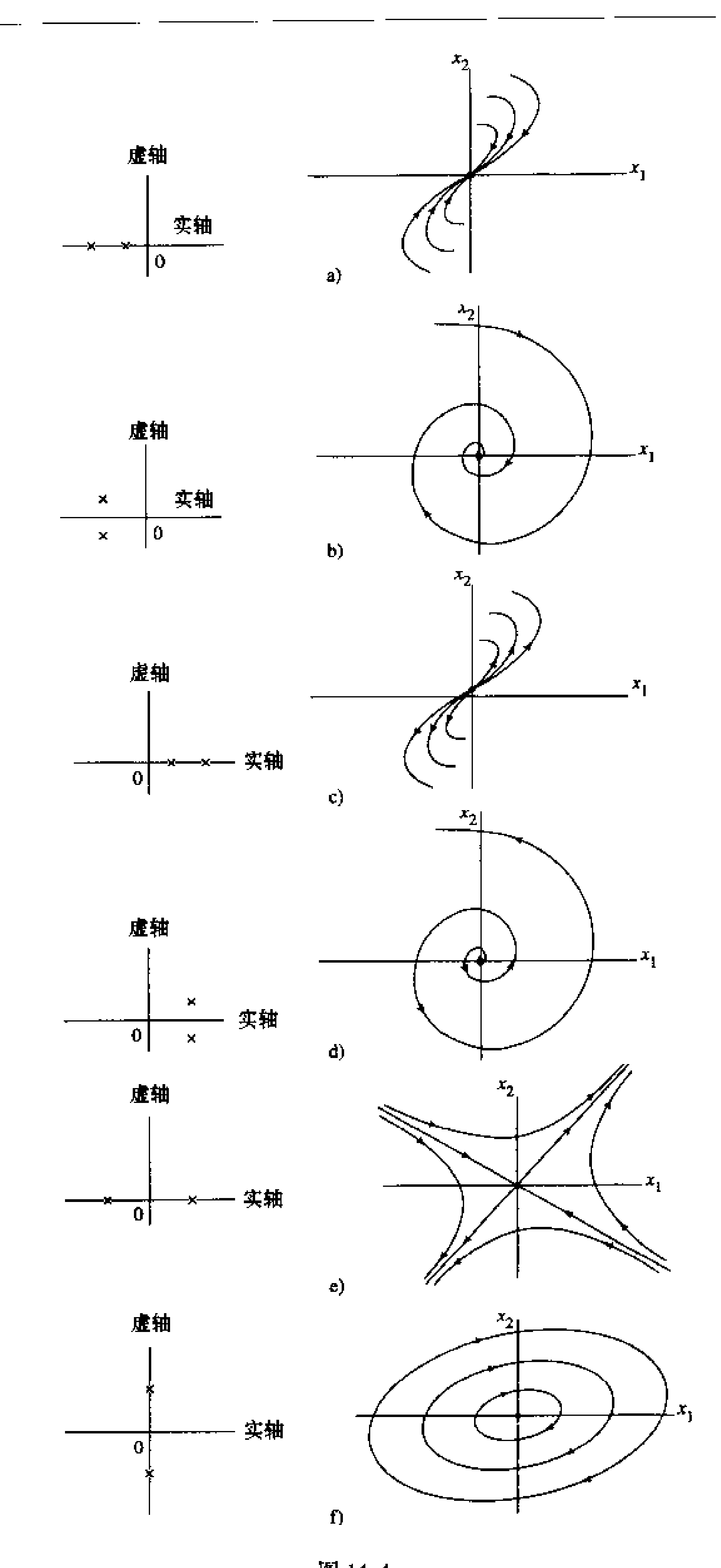


图 14-4

a)稳定结点 b)稳定焦点 c)不稳定结点 d)不稳定焦点 e)鞍点 f)中心

稳定性定义

就像已经简略叙述过的那样，状态空间方程的线性化可以提供关于一个平衡状态的局部稳定特性的有用信息。但是，为了能以一种更加细节化的方式研究非线性动态系统的稳定性，我们需要关于平衡状态的稳定性和收敛性的精确定义。

在和带有平衡状态 $\bar{\mathbf{x}}$ 的自治非线性动态系统相关的环境中，稳定性和收敛性的定义如下(Cook,1986)：

定义 1 若对于任意给定的正数 ϵ，存在一正数 δ，使得当满足条件 $\|\mathbf{x}(0)-\bar{\mathbf{x}}\|<\delta$ 时，对于所有 $t>0$ 恒有 $\|\mathbf{x}(t)-\bar{\mathbf{x}}\|<\epsilon$，则称平衡状态 $\bar{\mathbf{x}}$ 为一致稳定的。

这一定义表明如果初始状态 $\mathbf{x}(0)$很接近 $\bar{\mathbf{x}}$，则系统的一条轨线可能会停留在平衡状态 $\bar{\mathbf{x}}$ 很小的一个邻域内。

定义 2 如果存在一个正数 δ 使得当条件 $\|\mathbf{x}(0)-\bar{\mathbf{x}}\|<\delta$ 时，对于 $t\to\infty$ 有 $\mathbf{x}(t)\to\bar{\mathbf{x}}$，则称平衡状态 $\bar{\mathbf{x}}$ 为收敛的。

第二个定义的含义在于如果一条轨线的初始状态 $\mathbf{x}(0)$足够接近于平衡状态 $\bar{\mathbf{x}}$，则在时间 t 接近无穷的时候由状态向量 $\mathbf{x}(t)$ 所描述的轨线将收敛于 $\bar{\mathbf{x}}$。

定义 3 若平衡状态是稳定的并且是收敛的，则称平衡状态 $\bar{\mathbf{x}}$ 为渐近稳定的。

这里我们要注意稳定性和收敛性是互相独立的性质。只有两者都具备才有渐近稳定性。

定义 4 如果平衡状态是稳定的并且所有的系统轨线在时间 t 接近无穷的时候都收敛于 $\bar{\mathbf{x}}$，则称平衡状态 $\bar{\mathbf{x}}$ 为渐近稳定的或者全局渐近稳定的。

这一定义意味着系统不可能有其他的平衡状态，而且它要求系统中的每一条轨线对所有的时间 $t>0$ 都保持有界。换句话说，全局渐近稳定性意味着对于任意初始条件系统都将最终稳定在一个稳态上。 672

例 14.1 令由式(14.2)表示的非线性动态系统的解 $\mathbf{u}(t)$就像图 14-5 中说明的那样随时间变化。如图 14-5 所示，为了解 $\mathbf{u}(t)$是一致稳定的，我们需要 $\mathbf{u}(t)$和任何其他解 $\mathbf{v}(t)$在同样的 t 值(即时间"滴答")时保持互相接近。这种行为被称为两个解 $\mathbf{u}(t)$和 $\mathbf{v}(t)$的同步对应(isochronous correspondence)(E.A.Jackson,1989)。设解 $\mathbf{u}(t)$是收敛的，假定对于每一个其他的解 $\mathbf{v}(t)$，在 $t=0$ 处 $\|\mathbf{v}(0)-\mathbf{u}(0)\|\leqslant\delta(\epsilon)$成立，则解 $\mathbf{v}(t)$和 $\mathbf{u}(t)$当 t 趋于无穷时收敛于平衡状态。 ■

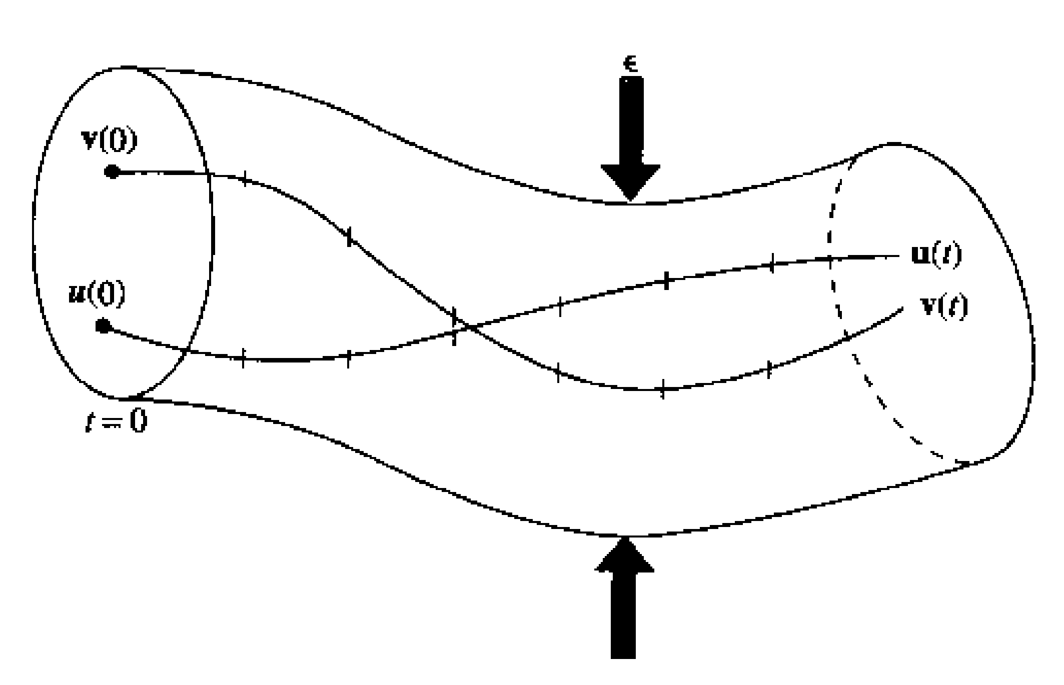


图 14-5 状态向量一致稳定(收敛)的概念图示

Lyapunov 定理

已经定义了动态系统的稳定性和渐近稳定性，下一个要考虑的问题就是确定稳定性。显而易见我们可以通过实际地找到系统状态空间方程的所有可能解来做到；但是这种方法即使不是不可能也是非常困难的。一个更精致的方法可以在现代稳定性理论中找到，该理论由 Lyapunov 创立。具体地，我们可以通过应用 Lyapunov 直接方法来研究稳定性问题，这个方法使用叫做 Lyapunov 函数的状态向量的连续标量函数。

由方程(14.2)描述的具有状态向量 $\mathbf{x}(t)$和平衡状态 $\bar{\mathbf{x}}$ 的自治非线性动态系统，关于它的状态空间的稳定性和渐近稳定性的 Lyapunov 定理可以陈述如下：

定理1 如果在 $\bar{\mathbf{x}}$ 的小邻域内存在一个正定函数 $V(\mathbf{x})$，其对时间的导数在该区域内是半负定的，则平衡状态 $\bar{\mathbf{x}}$ 是稳定的。

定理2 如果在 $\bar{\mathbf{x}}$ 的小邻域内存在一个正定函数 $V(\mathbf{x})$，其对时间的导数在该区域内是负定的，则平衡状态 $\bar{\mathbf{x}}$ 是渐近稳定的。

673

满足以上要求的标量函数 $V(\mathbf{x})$叫做平衡状态的 $\bar{\mathbf{x}}$ 的 Lyapunov 函数。

这两个定理要求 Lyapunov 函数是正定函数。这样的函数定义如下：在状态空间$\mathscr{L}$中，如果对所有的 $\mathbf{x}\in\mathscr{L}$，满足以下要求，则称其为正定函数。

1. 函数 $V(\mathbf{x})$对状态向量 $\mathbf{x}$ 中所有元素有连续偏导数
2. $V(\bar{\mathbf{x}})=0$
3. 如果 $\mathbf{x}\neq\bar{\mathbf{x}}$，则 $V(\mathbf{x})>0$

给出这样的 Lyapunov 函数 $V(\mathbf{x})$，根据定理 1，若[2] 对于 $\mathbf{x}\in\mathscr{U}-\bar{\mathbf{x}}$

$$\frac{d}{dt}V(\mathbf{x})\leqslant 0 \qquad 对于\ \mathbf{x}\in\mathscr{U}-\bar{\mathbf{x}} \tag{14.11}$$

成立，则平衡状态 $\bar{\mathbf{x}}$ 是稳定的，其中$\mathscr{U}$是 $\bar{\mathbf{x}}$ 的小邻域。此外，根据定理 2，若

$$\frac{d}{dt}V(\mathbf{x})<0 \qquad 对于\ \mathbf{x}\in\mathscr{U}-\bar{\mathbf{x}} \tag{14.12}$$

成立，则平衡状态 $\bar{\mathbf{x}}$ 是渐近稳定的。

这一讨论的重要之处在于可以不求解系统的状态空间方程而直接应用 Lyapunov 定理。不幸的是，定理并没有给出如何找到 Lyapunov 函数的提示；在每种情况它是一件创造性、尝试和犯错误的事情。对于感兴趣的很多问题，能量函数可以起到 Lyapunov 函数的作用。但是，无法找到适用的 Lyapunov 函数并不能证明系统的不稳定性。因为 Lyapunov 函数的存在是系统稳定的充分条件，而不是必要条件。

Lyapunov 函数 $V(\mathbf{x})$为对由式(14.2)描述的非线性动态系统进行全局稳定性分析提供数学基础。另一方面，基于 Jacobi 矩阵 $\mathbf{A}$，使用式(14.10)为进行系统局部稳定性分析提供基础。全局稳定性分析的结论比局部分析更有力；因为每个全局稳定的系统必定是局部稳定的，反之则不然。

14.4 吸引子

耗散系统一般可以用存在吸引集或者比状态空间维数低的流形来表征。“流形”是指嵌入在 N 维状态空间中的一个 k 维曲面，它由方程组

$$M_j(x_1, x_2, \cdots, x_N) = 0, \quad \begin{cases} j = 1,2,\cdots,k \\ k < N \end{cases} \tag{14.13}$$

674

定义，其中$x_1, x_2, \cdots, x_N$是系统N维状态向量的元素，M_j是这些元素的一个函数。这些流形称为*吸引子*[3]，这是因为吸引子为有界子集，初始条件为非零状态空间体积的区域随时间增加而收敛到它们(Ott,1993)。

流形可以是状态空间中的一个点，这种情况叫做*点吸引子*。另外，它也可以是周期性轨道，这种情况叫做稳定的*极限环*，稳定意味着附近的轨线渐近地趋近它。图14-6描绘这两种类型的吸引子。吸引子代表动态系统中的惟一可以通过用实验方法观察到的*平衡状态*。但是，注意在吸引子的情况下，平衡状态(equilibrium)既不意味着一个静态平衡(static equilibrium)，也不意味一个定常状态(steady state)。例如，一个极限环代表一个吸引子的稳定状态(stable state)，但是它随时间连续变化。

在图14-6中，我们注意每个吸引子由它自己独有的区域包围。这样的区域叫做*吸引盆(域)*(basin(domain) of attraction)。同时注意系统的每个初始状态都在某一吸引子的盆中。分隔不同吸引盆的边界叫做*分界线*(separatrix)。图14-6中盆的边界由轨线T_1、鞍点Q和轨线T_2的并表示。

极限环组成非线性系统的平衡点变得不稳定时出现的振荡行为的典型形式。因此，它可能出现在任意阶的系统中。虽然如此，极限环是二阶系统特殊的特征。

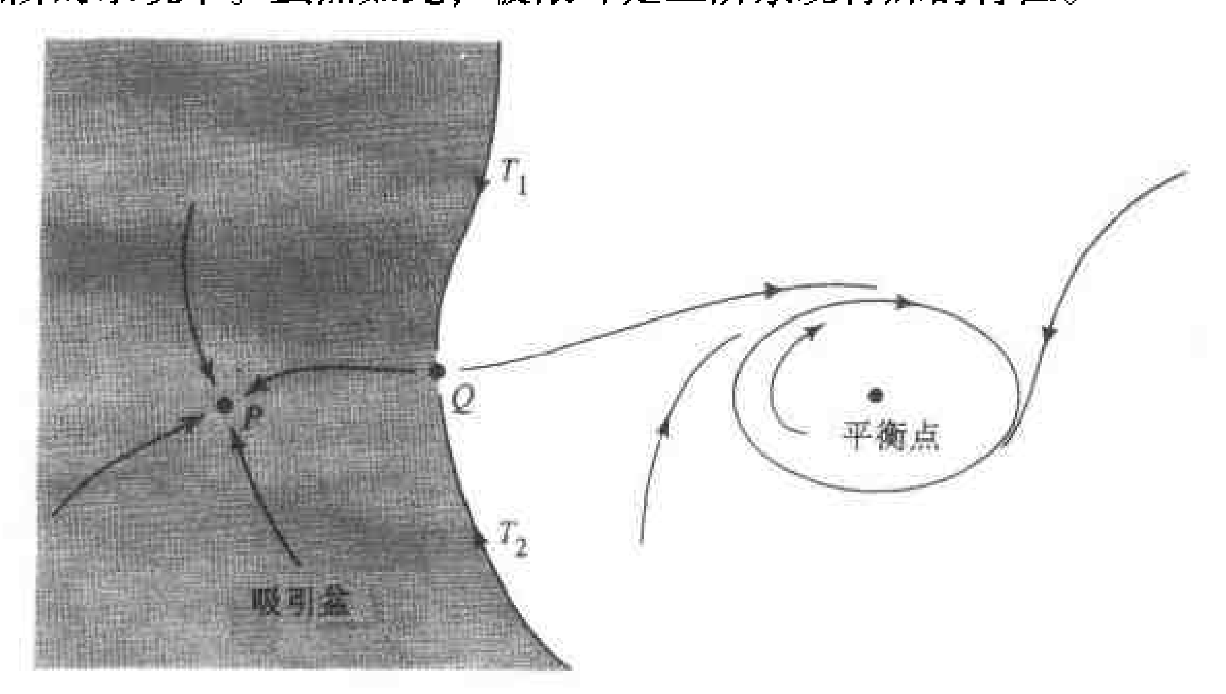


图14-6 吸引盆概念和分界线思想的说明图

双曲吸引子

考虑一个点吸引子，通过使用14.2节中描述的方式将它的非线性动态方程在平衡状态$\bar{\mathbf{x}}$附近线性化。令$\mathbf{A}$表示系统在$\mathbf{x}=\bar{\mathbf{x}}$处计算出的Jacobi矩阵。如果$\mathbf{A}$所有特征值的绝对值都小于1，则吸引子是*双曲吸引子*(hyperbolic attractor)(Ott,1993)。例如，二阶双曲吸引子的流可以为图14-4a或者14-4b中所显示的形式；两种情况下Jacobi矩阵$\mathbf{A}$的特征值都有负实数部分。双曲吸引子在称为消除梯度问题的研究中受到特别的关注，这种问题出现在动态驱动的递归网络中；这一问题在下一章讨论。

675

14.5 神经动态模型

对非线性动态系统的性能有所了解之后，准备在本节和下一节探讨一下神经动力学所包

含的一些重要问题。我们要强调的是，对于神经动力学还没有一个被普遍认可的定义。我们也不是要给出这样一个定义，而是将定义本章中所考虑的神经动力学最普遍的属性。特别地，讨论将局限于状态变量是连续的并且运动方程由微分方程或差分方程描述的神经动态系统。受关注的系统具有四个普遍特性(Peretto and Niez, 1986; Pineda, 1988a)：

1. 大量自由度。人脑皮层是高度并行的分布式系统，据估计大约有100亿个神经元，每个神经元用一个或更多状态变量描述。据信这样一个神经动力学系统的计算能力和容错能力是系统的集体动力学的结果。系统可以表征为大量的由每个突触连接的强度(效能(efficacy))表示的耦合常量。

2. 非线性性。神经动力学系统是非线性的。事实上，非线性是建立通用计算机器的基础。

3. 耗散性。神经动力学系统是耗散的。因此，它由状态空间体积随时间的延展收敛于一低维流形这一收敛性表征。

4. 噪声。最后，噪声是神经动态系统内在特征。在实际神经元中，膜噪声在突触连接处产生(Katz, 1966)。

噪声的存在需要对神经元行为利用概率处理，这给分析神经动力学系统增加了另一层次上的复杂性。对随机神经动力学的详细处理超出本书的范围。因此，以后的材料中均忽略噪声的影响。

加性模型

考虑图14-7中所显示的神经元的无噪声动态模型，其数学基础已在13章讨论过了。使用物理术语，突触权值$w_{j1}, w_{j2}, \cdots, w_{jN}$表示传导系数，各自的输入$x_1(t), x_2(t), \cdots, x_N(t)$表示电压，$N$是输入数量。这些输入被用于有如下特点的电流求和连接上：

676

- 低输入阻抗
- 单位电流增益
- 高输出阻抗

因此对输入电流来说，它扮演求和节点的角色。图14-7中非线性元素(激活函数)流向输入节点的总电流流量为

$$\sum_{i=1}^{N} w_{ji}x_i(t) + I_j$$

其中第一项(求和项)是由于刺激$x_1(t), x_2(t), \cdots, x_N(t)$分别作用在突触权值(传导系数)$w_{j1}, w_{j2}, \cdots, w_{jN}$上，第二项是由于电流源$I_j$代表额外施加的偏置。令$v_j(t)$表示非线性激活函数$\varphi(\cdot)$输入处的诱导局部域。因此我们可以表示从非线性元素的输入节点流出的总电流量为

$$\frac{v_j(t)}{R_j} + C_j\frac{dv_j(t)}{dt}$$

其中第一项是由于漏泄阻抗R_j，第二项是由于漏泄电容C_j。根据Kirchoff电流定律，我们知道电路中流向任何节点的总电流流量为零。通过应用Krichoff电流定律于图14-7中的非线性输入节点，得到

$$C_j \frac{dv_j(t)}{dt} + \frac{v_j(t)}{R_j} = \sum_{i=1}^{N} w_{ji}x_i(t) + I_j \tag{14.14}$$

式(14.14)左端的电容项 $C_j dv_j(t)/dt$ 是在神经元模型上添加动力学(记忆)的最简单的途径。给定诱导局部域 $v_j(t)$，可以通过使用非线性关系

$$x_j(t) = \varphi(v_j(t)) \tag{14.15}$$

确定神经元 j 的输出。由式(14.14)描述的 RC 模型通常称为加性模型；这一术语用于区别本模型和 w_{ji} 依赖于 x_i 的乘法(或并联)模型(Grossberg, 1982)。 677

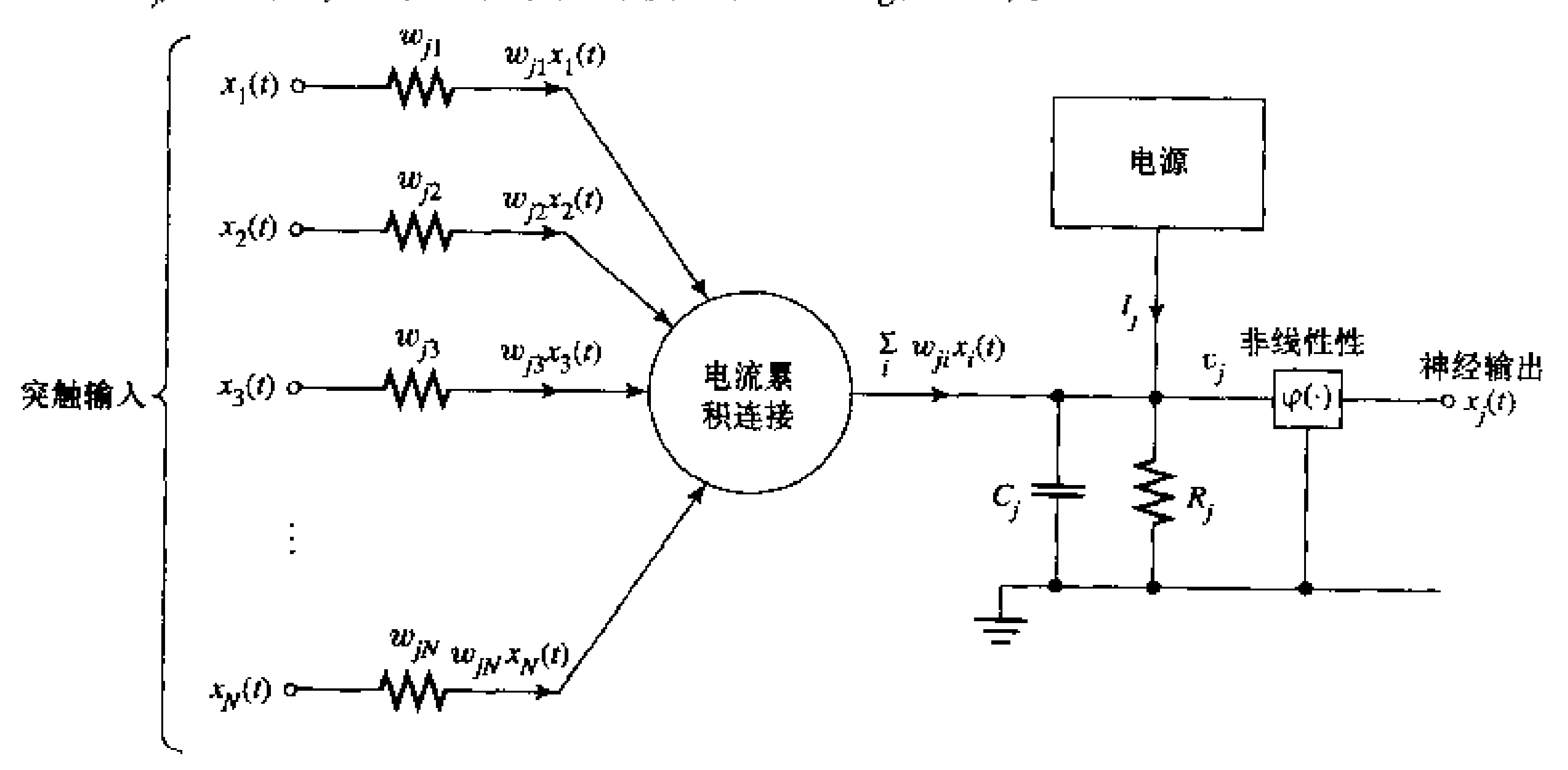


图 14-7 神经元的加性模型

由式(14.14)描述的加性模型的一个显著特性就是相邻神经元 i 施加在神经元 j 上的信号 $x_i(t)$ 是随时间 t 缓慢改变的。因此描述的模型组成传统神经动力学的基础[4]。

继续考虑一个包含 N 个互相连接的神经元的递归网络，假设其中每一个神经元都有由式(14.14)和式(14.15)描述的同样数学模型。那么，忽略神经元内部时间传播的延迟，我们可以用联立的一阶微分方程组

$$C_j \frac{dv_j(t)}{dt} = -\frac{v_j(t)}{R_j} + \sum_{i=1}^{N} w_{ji}x_i(t) + I_j, j = 1,2,\cdots,N \tag{14.16}$$

的系统定义网络的动力学，它和状态方程(14.1)有同样的数学形式，并且是式(14.14)中各项的简单再排列。假设和神经元 j 的输出 $x_j(t)$ 相关的激活函数 $\varphi(\cdot)$ 对它的诱导局部域来说是连续和可微的函数。普遍使用的激活函数是 logistic 函数

$$\varphi(v_j) = \frac{1}{1 + \exp(-v_j)}, j = 1,2,\cdots,N \tag{14.17}$$

14.6 节至 14.11 节中描述的学习算法存在的必要条件在于由式(14.15)和(14.16)描述的递归网络具有固定点(即点吸引子)。

相关模型

为了简化说明，我们假设式(14.16)中神经元 j 的时间常数 $\tau_j = R_jC_j$ 对所有的 j 都是一样的。那么，通过关于这一时间常数的公共值归一化时间 t，并关于 R_j 归一化 w_{ji} 和 I_j，可以

重新构造式(14.16)的模型如下：

$$\frac{dv_j(t)}{dt} = -v_j(t) + \sum_i w_{ji}\varphi(v_i(t)) + I_j, j = 1,2,\cdots,N \tag{14.18}$$

其中我们也并入了式(14.15)。联立一阶非线性微分方程组(14.18)的吸引子结构和以下描述的紧密相关模型的吸引子结构基本上相同(Pineda, 1987)：

678

$$\frac{dx_j(t)}{dt} = -x_j(t) + \varphi\left(\sum_i w_{ji}x_i(t)\right) + K_j, \quad j = 1,2,\cdots,N \tag{14.19}$$

由式(14.18)描述的加性模型中，独立神经元的诱导局部域$v_1(t), v_2(t), \cdots, v_N(t)$构成状态向量。另一方面，在由式(14.19)描述的相关模型中，神经元的输出$x_1(t)$, $x_2(t), \cdots, x_N(t)$构成状态向量。

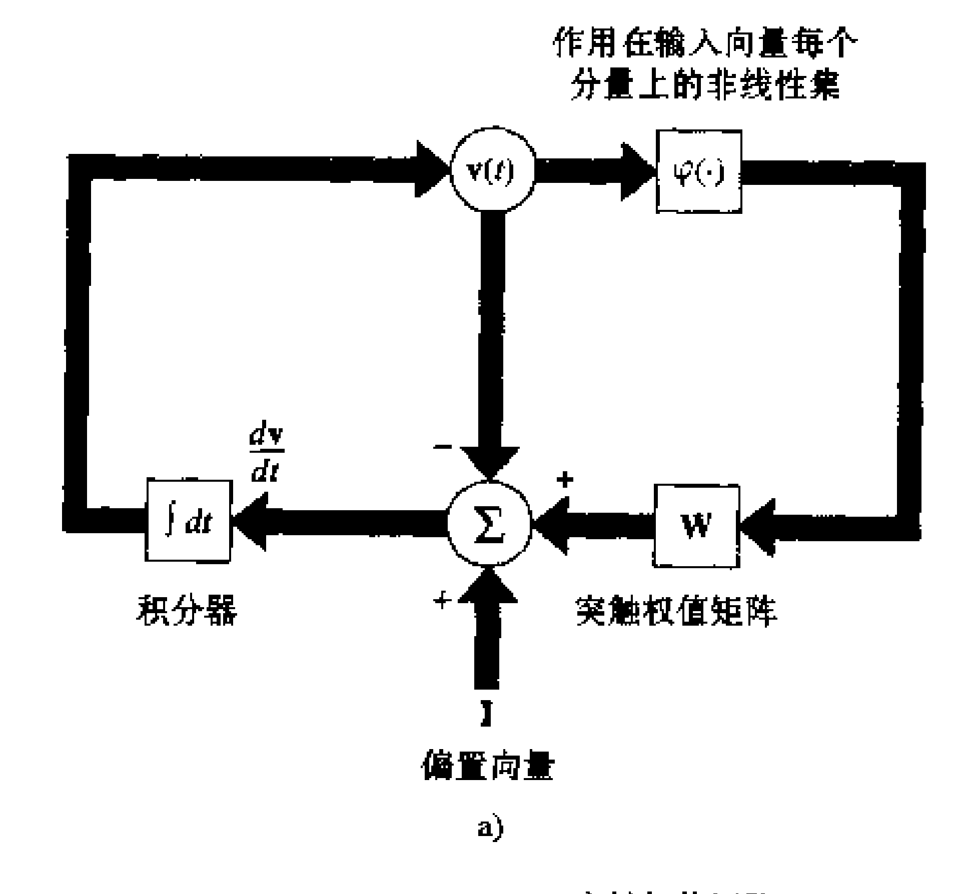
这两种神经动力学模型事实上通过线性的可逆变换是相关的。具体地，通过在式(14.19)两侧同乘以w_{kj}，对j求和，然后用变换

$$v_k(t) = \sum_j w_{kj}x_j(t)$$

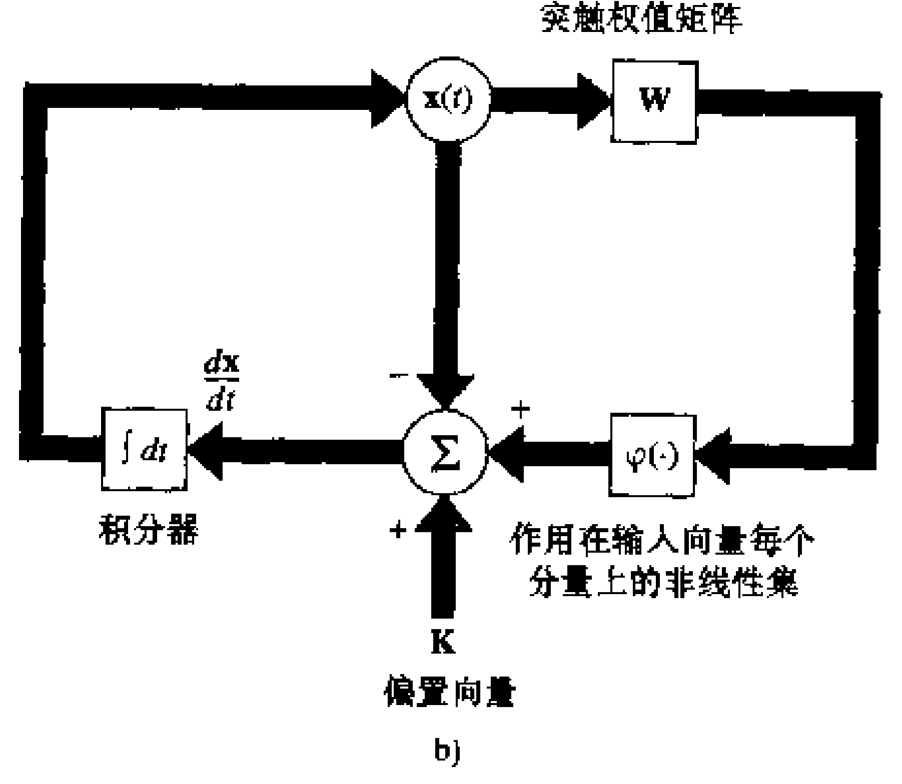
进行替换，得到一个由式(14.18)所描述的类型的模型，并且由此发现两个模型的偏置项由

$$I_k = \sum_j w_{kj}K_j$$

相关联。这里重要之处是注意与式(14.18)的加性模型的稳定性相关的结果也适用于与式(14.19)相关的模型。



a)



b)

图 14-8
a) 由联立一阶微分方程组(14.18)表示的神经动态系统框图 b) 由方程组(14.19)描述的相关模型的框图

这里描述的两种神经动力学模型之间的紧密关系也可以用图14-8中的框图来说明。图中a和b部分分别对应于式(14.18)和(14.19)的矩阵公式；**W**是突触权值矩阵，**v**(t)是在时间t的诱导局部域向量，**x**(t)是在时间t的神经元输出向量。两种模型中反馈的存在图14-8中是清晰可见的。

679

14.6 作为递归网络范例的吸引子操作

当神经元数量N非常大的时候，除去噪声的影响，式(14.16)描述的神经动力学模型具有14.5节中概述的普遍特性：大量的自由度、非线性性和耗散性。因而，这样一个神经动

力学模型可能拥有复杂的吸引子结构并因此展示出有用的计算能力。

确认具有计算对象(如联想记忆、输入-输出映射器)的吸引子是神经网络范例的一个基础。为了实现这一思想，我们必须训练控制吸引子在系统状态空间中的位置。于是为了以希望的形式编码信息或者学习感兴趣的时间结构，学习算法采用了非线性动力学方程的形式操纵吸引子在状态空间的位置。通过这一途径，在机器的物理性能和计算的算法之间建立紧密的联系是可能的。

利用神经网络的集体属性实现计算任务的一种途径就是经由能量最小化的概念。在14.7节和14.10节中将分别考虑的Hopfield网络和盒中脑状态模型是这种方法著名的例子。这两种模型都是能量最小化网络；它们的不同之处在于应用领域不同。Hopfield网络作为按内容寻址存储或者用于解决组合类型最优化问题的模拟计算机是有用的。另一方面，盒中脑状态模型对于聚类类型的应用是有用的。本章后面几节将对这些应用进行说明。

Hopfield网络和盒中脑状态模型是不含隐藏神经元的联想记忆的实例；联想记忆是智能行为的一个重要来源。另一个神经动力学模型是输入输出映射器类型的，它的运行依赖于隐藏神经元的可用性。在这后一种情况中，最速下降方法经常被用于最小化根据网络参数定义的代价函数，并因此改变吸引子位置。这后一种神经动力学模型的应用以在下一章中讨论的动态驱动递归网络的作为例子。

14.7 Hopfield模型

如图14-9中描绘的那样，Hopfield网络(模型)包含一组神经元和一组相应的单位延迟，构成一个多回路反馈系统。反馈回路的数量等于神经元数量。基本上，每个神经元的输出都通过一个单位延迟元素被反馈到网络中另外的每一个神经元。换句话说，网络中没有自反馈；避免使用自反馈的原因将在后面解释。

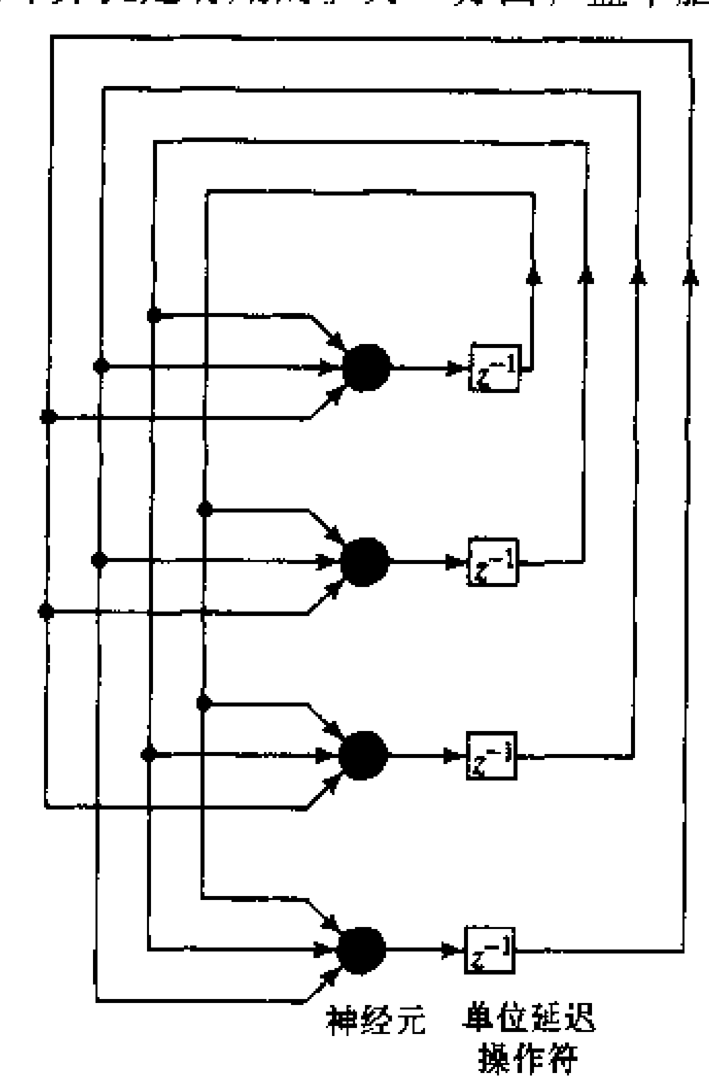


图14-9 包含4个神经元的Hopfield网络结构图

680

为了研究Hopfield网络的动力学，我们使用式(14.16)描述的基于神经元加性模型的神经动力学模型。

认识到 $x_i(t)=\varphi_i(v_i(t))$ 之后，我们可以把式(14.16)改写成以下形式：

$$C_j\frac{d}{dt}v_j(t)=-\frac{v_j(t)}{R_j}+\sum_{i=1}^{N}w_{ji}\varphi_i(v_i(t))+I_j,j=1,\cdots,N \qquad (14.20)$$

为了继续讨论，我们作出以下假定：

1. 突触权值矩阵是对称的，表示为

$$w_{ji}=w_{ij} \qquad \text{对所有 } j \text{ 和 } i \qquad (14.21)$$

2. 每个神经元有它自己的非线性激活函数——因此在式(14.20)中使用 $\varphi_i(\cdot)$。

3. 非线性激活函数可逆，因此可以写成

$$v = \varphi_i^{-1}(x) \tag{14.22}$$

令 sigmoid 函数 $\varphi_i(v)$ 由双曲线正切函数

$$x = \varphi_i(v) = \tanh\left(\frac{a_i v}{2}\right) = \frac{1 - \exp(-a_i v)}{1 + \exp(-a_i v)} \tag{14.23}$$

定义，在原点处有 $a_i/2$ 的斜率，表示为

681

$$\frac{a_i}{2} = \left.\frac{d\varphi_i}{dv}\right|_{v=0} \tag{14.24}$$

此后我们将把 a_i 称为神经元 i 的增益。

因此，式(14.22)的逆输出 - 输入关系可以写成

$$v = \varphi_i^{-1}(x) = -\frac{1}{a_i}\log\left(\frac{1-x}{1+x}\right) \tag{14.25}$$

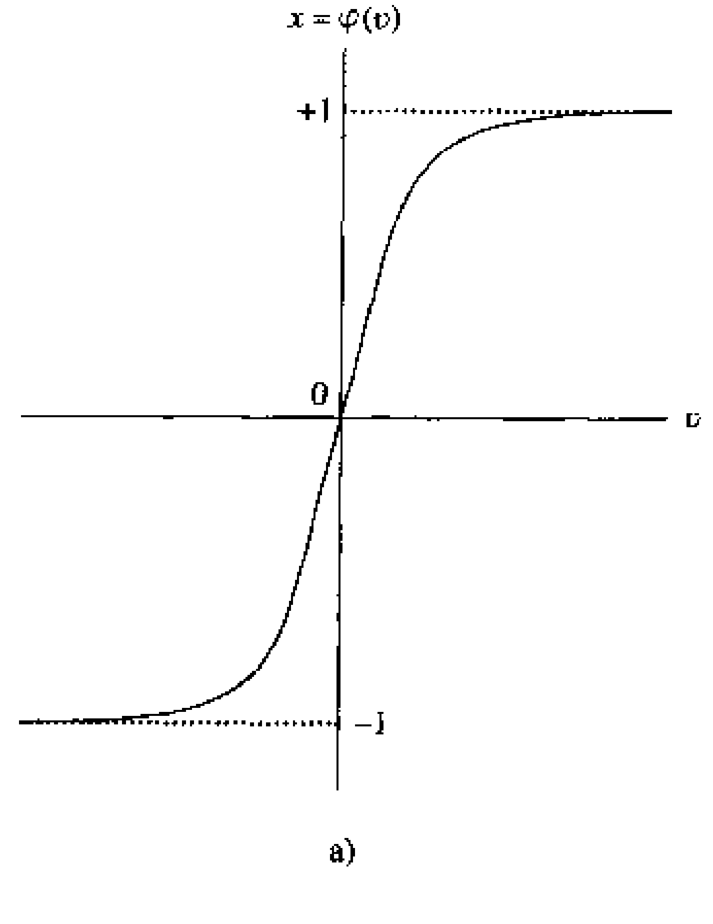
一个单位增益神经元的逆输出 - 输入关系的标准形式定义为

$$\varphi^{-1}(x) = -\log\left(\frac{1-x}{1+x}\right) \tag{14.26}$$

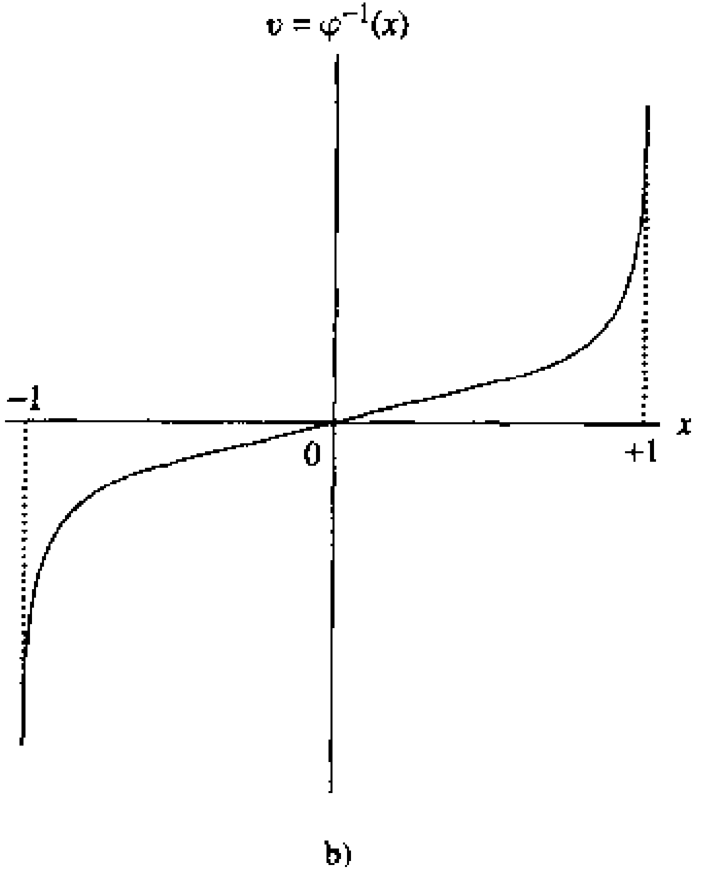
按照这一标准关系可以把式(14.25)改写为

$$\varphi_i^{-1}(x) = \frac{1}{a_i}\varphi^{-1}(x) \tag{14.27}$$

图 14-10a 显示标准 sigmoid 的非线性函数 $\varphi(v)$ 的曲线，图 14-10b 显示相应的非线性反函数 $\varphi^{-1}(x)$ 的曲线。



a)



b)

图 14-10
a)标准的 sigmoid 非线性图 b)它的逆

图 14-9 中的 Hopfield 网络的能量(Lyapunov)函数定义为(Hopfield, 1984)

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji}x_i x_j + \sum_{j=1}^{N}\frac{1}{R_j}\int_0^{x_j}\varphi_j^{-1}(x)\,dx - \sum_{j=1}^{N} I_j x_j \tag{14.28}$$

由式(14.28)定义的能量函数 E 为可能具有很多极小点的复杂图像。网络的动力学由寻找那些极小点的机制描述。

因此，求 E 对时间的微分，得到

$$\frac{dE}{dt} = -\sum_{j=1}^{N}\left(\sum_{i=1}^{N} w_{ji}x_i - \frac{v_j}{R_j} + I_j\right)\frac{dx_j}{dt} \tag{14.29}$$

由于神经动力学方程(14.20)所具有的特点，式(14.29)右端圆括号内的值被认为是 $C_j dv_j/dt$。于是可以把式(14.29)简化为

$$\frac{dE}{dt} = -\sum_{j=1}^{N} C_j\left(\frac{dv_j}{dt}\right)\frac{dx_j}{dt} \tag{14.30}$$

现在考虑由 x_j 定义的 v_j 的逆关系。将式(14.22)代入式(14.30)，得到

$$\frac{dE}{dt} = -\sum_{j=1}^{N} C_j\left[\frac{d}{dt}\varphi_j^{-1}(x_j)\right]\frac{dx_j}{dt} = -\sum_{j=1}^{N} C_j\left(\frac{dx_j}{dt}\right)^2\left[\frac{d}{dx_j}\varphi_j^{-1}(x_j)\right] \tag{14.31}$$

从图 14-10b 中可以看出逆输出输入关系 $\varphi_j^{-1}(x_j)$对输出 x_j 是单调增函数。因此它遵守

$$\frac{d}{dx_j}\varphi_j^{-1}(x_j) \geqslant 0 \qquad \text{对所有的 } x_j \tag{14.32}$$

682

我们也注意

$$\left(\frac{dx_j}{dt}\right)^2 \geqslant 0 \qquad \text{对所有的 } x_j \tag{14.33}$$

因而，所有在式(14.31)右端求和的因子都是非负的。换句话说，对式(14.28)定义的能量函数 E 来说，我们有$\frac{dE}{dt} \leqslant 0$。由式(14.28)的定义看出函数 E 是有界的。因此，我们可以作出以下两个陈述：

683

1. 能量函数 E 是连续 Hopfield 模型的 Lyapunov 函数。
2. 根据 Lyapunov 定理 1 模型是稳定的。

换句话说，由非线性一阶微分方程组(14.20)的系统描述的连续 Hopfield 模型的时间演化代表状态空间中的一条轨线，该轨线找出能量(Lyapunov)函数 E 的极小值并在这样的固定点上终止。从式(14.31)也要注意，仅当

$$\frac{d}{dt}x_j(t) = 0 \qquad \text{对所有 } j$$

时，导数 dE/dt 变为零。因此可以进一步写出

$$\frac{dE}{dt} < 0 \qquad \text{除在一个固定之外} \tag{14.34}$$

式(14.34)给出了下述定理的基础：

Hopfield 网络的(Lyapunov)能量函数 E 是时间的单调减函数。

因此，Hopfield 网络是全局渐近稳定的；吸引子固定点是能量函数的极小值，反之亦然。

离散和连续 Hopfield 模型的稳定状态之间的关系

Hopfield 网络可以用连续方式或离散方式运行，依赖于描述神经元所采用的模型。连续模型的运行基于前面描述的加性模型。另一方面，离散模型的运行基于 McCulloch-Pitts 模型。通过重新定义神经元的输入－输出关系，很容易在连续 Hopfield 模型稳定状态和相应的离散 Hopfield 模型的稳定状态之间建立联系，使得这样的关系满足下面两个简化特性：

1. 神经元的输出有渐近值

$$x_j = \begin{cases} +1, & v_j = \infty \\ -1, & v_j = -\infty \end{cases} \tag{14.35}$$

2. 神经元激活函数的中点在原点处，表示为

$$\varphi_j(0) = 0 \tag{14.36}$$

684

相应地，可以对所有的 j 设置偏置 I_j 为零。

为了表示连续 Hopfield 模型的能量函数 E，允许神经元有自反回路。另一方面，离散 Hopfield 模型不需要自反回路。因此，可以通过在两种模型中对所有的 j 都设置 $w_{jj}=0$ 来简

化讨论。

根据这些观察，可以用如下形式重新定义式(14.28)给出的连续 Hopfield 模型的能量函数：

$$E = -\frac{1}{2}\sum_{\substack{i=1}}^{N}\sum_{\substack{j=1\\ i\neq j}}^{N} w_{ji}x_i x_j + \sum_{j=1}^{N}\frac{1}{R_j}\int_0^{x_j}\varphi_j^{-1}(x)\,dx \tag{14.37}$$

由式(14.27)定义反函数 $\varphi_j^{-1}(x)$。于是可以重写式(14.37)的能量函数如下：

$$E = -\frac{1}{2}\sum_{\substack{i=1}}^{N}\sum_{\substack{j=1\\ i\neq j}}^{N} w_{ji}x_i x_j + \sum_{j=1}^{N}\frac{1}{a_j R_j}\int_0^{x_j}\varphi^{-1}(x)\,dx \tag{14.38}$$

积分

$$\int_0^{x_j}\varphi^{-1}(x)\,dx$$

有图14-11中显示的标准形式。在 $x_j=0$ 积分值为零，其他情况其值为正。假设在 x_j 接近 ± 1 时其值非常大。但是，如果神经元 j 增益 a_j 变为无穷大(例如 sigmoid 函数的非线性趋于理想的硬限制形式)，式(14.38)中的第二项就小得可以忽略不记了。在限制情况下，对所有的 j，当 $a_j=\infty$ 时连续 Hopfield 模型的极大、极小值变成和离散 Hopfield 模型中的对应值相等。后一情况下，能量(Lyapunov)函数的定义简化为

685

$$E = -\frac{1}{2}\sum_{\substack{i=1}}^{N}\sum_{\substack{j=1\\ i\neq j}}^{N} w_{ji}x_i x_j \tag{14.39}$$

其中第 j 个神经元状态为 $x_j=\pm 1$。因此，我们得出结论：高增益的、连续的和确定的 Hopfield 模型仅有的稳定点对应于离散随机 Hopfield 模型的稳定点。

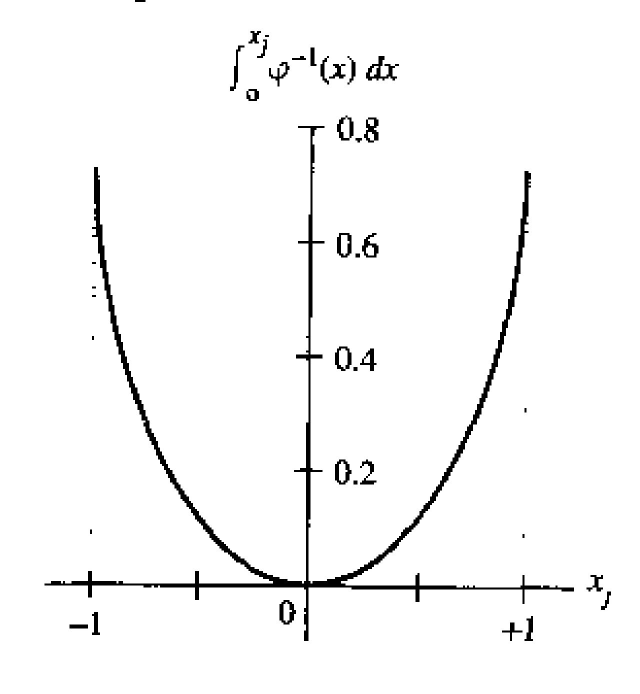


图14-11 积分 $\int_0^{x_j}\varphi^{-1}(\mathbf{x})\,d\mathbf{x}$ 的图形

然而，当每一个神经元 j 有很大但是有限的增益 a_j 时，我们发现式(14.38)右端第二项对连续模型的能量函数有明显的贡献。特别，这一贡献在靠近定义模型状态空间的超立方体的所有面、边和角点处都很大并且为正。而另一方面，该贡献在远离曲面的点处又小得可以忽略。因此，这种模型能量函数的最大值在角点处，但最小值却略微向超立方体的内部偏移(Hopfield, 1984)。

图14-12画出两个神经元的连续 Hopfield 模型的能量等值线图或能量图。两个神经元的输出定义图中的两个坐标轴。图14-12中左下角和右上角代表无穷增益限制情况下的稳定最小值；有限增益情况下的最小值将向内部偏移。流向固定点(即稳定最小值)的流可以解释为式(14.28)定义的能量函数 E 的最小化的解。

离散 Hopfield 模型作为按内容寻址存储器

Hopfield 网络作为按内容寻址存储器(content-addressable memory)在文献中吸引了人们巨大的注意。在这一应用领域，我们预先知道网络的固定点，它们对应被存储模式。但是，产生期望中固定点的网络突触权值是未知的，因而问题在于如何确定它们。按内容寻址存储器的主要功能是根据模式不完整或有噪声的表示获取存储在存储器中相应模式(项)。为了以简洁

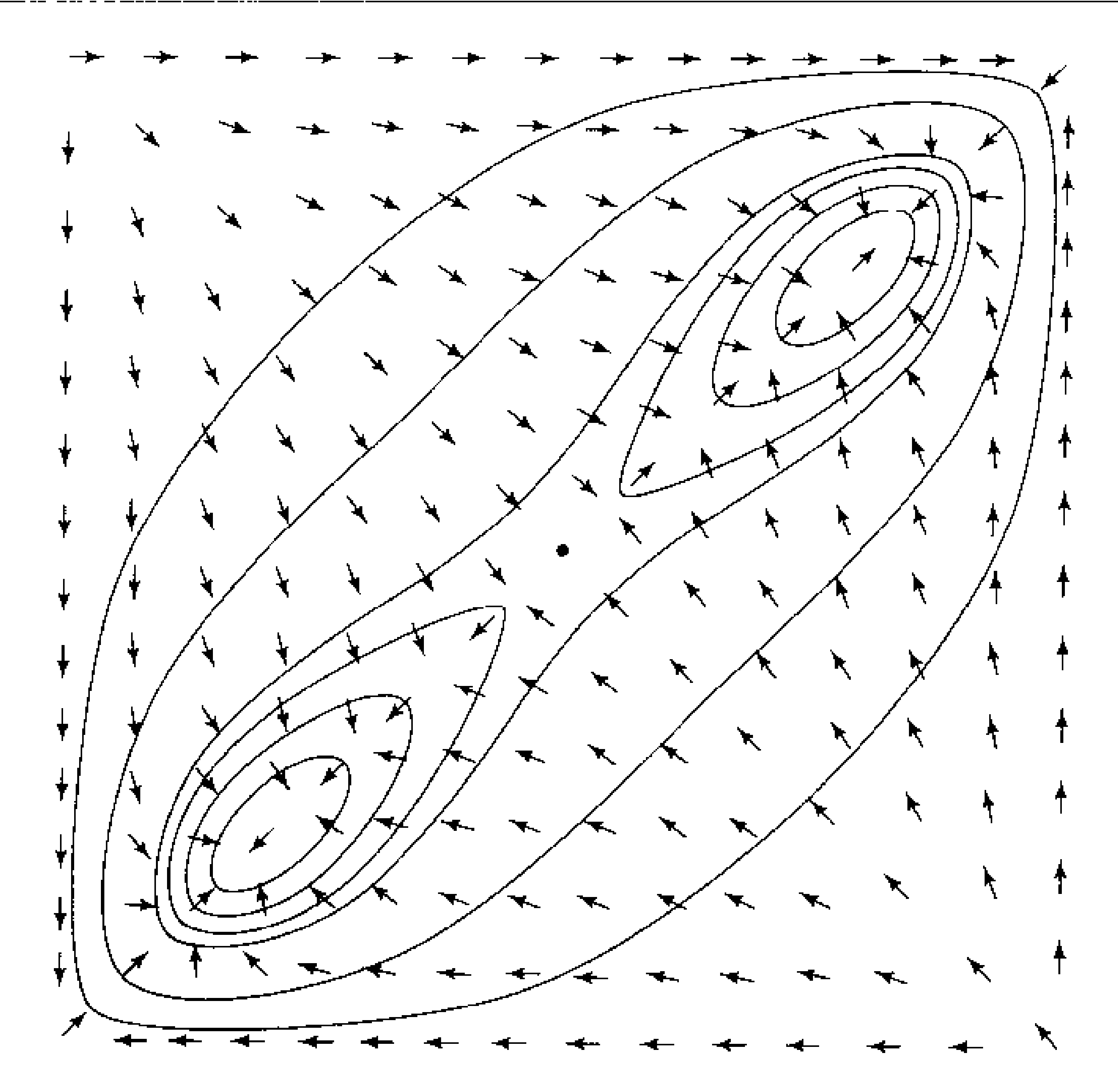


图 14-12 两个神经元的双稳定态系统的能量等值线图。纵轴和横轴为两个神经元的输出。稳定状态位于左下角和右上角，不稳定的极点位于另外两个角。箭头表示状态的移动。移动一般不垂直于能量的等值线图。（经美国国家科学院允许，摘自 J.J.Hopfield，1984）

686

方式说明这一陈述的含义，最好的方法就是引用 Hopfield 1982 年的论文：

假定存储在存储器中的项是"H.A.Kramers & G.H.Wannier *Physi Rev*.*60*，252(1941)."一个普通的按内容寻址存储器，根据足够的部分信息能检索这个完整的存储项。输入"& Wannier，(1941)"可能就足够了。理想的存储器能处理错误并且甚至只输入"Wannier，(1941)"就能检索这一参考文献。

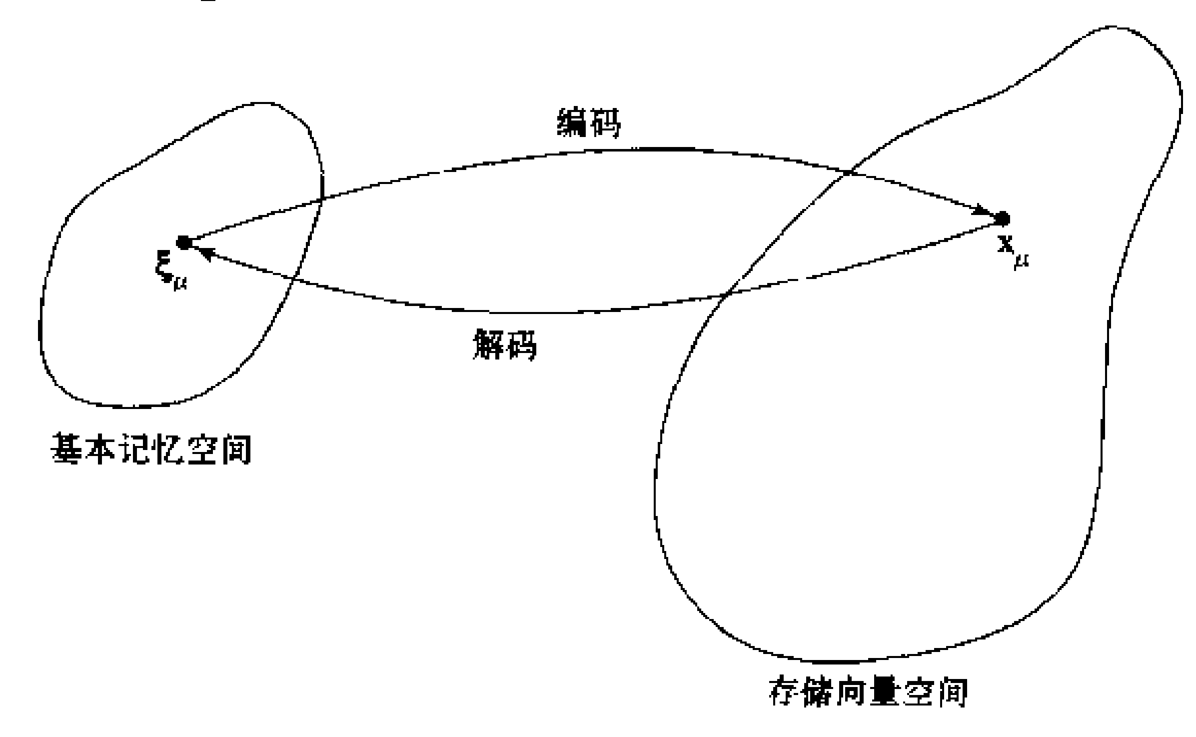
因此，按内容寻址存储器的一个重要属性就是在给出存储模式的信息内容的一个合理子集的情况下检索该模式的能力。此外，根据提供的线索能够覆盖不一致的信息，在这种意义下按内容寻址存储是可以纠错的。

按内容寻址存储器(CAM)的本质是映射基本存储 ξ_μ 到动态系统的固定点(稳定点)$\mathbf{x}_\mu$ 上，就像图 14-13 描绘的那样。在数学可以把这个映射表示为

$$\xi_\mu \rightleftharpoons \mathbf{x}_\mu$$

的形式。从左向右的箭头代表编码操作，而从右向左的箭头代表解码操作。网络状态空间的吸引子固定点为网络的基本记忆或做原型状态。假设现在网络被呈现给一个模式，这个模式

包含基本记忆的部分但足够的信息。那么我们可以将该特定模式表示为状态空间中的起点。原则上，倘若该起点靠近表示待检索记忆的固定点(即它位于固定点的吸引盆内部)，则系统应该随时间演化并最终收敛于记忆状态本身。在那个点上全部的记忆由网络生成。结果Hopfield网络有再现(emergent)的性质，该性质帮助它检索信息和处理错误。



687

图14-13 递归网络实现的编码－解码示意图

在使用McCulloch and Pitts(1943)的正规神经元作为基本处理单元的Hopfield模型中，每一个这样的神经元具有由作用其上的诱导局部域所决定的两个状态。神经元 i 的"开"或"点火"状态用输出值 $x_i=+1$ 表示，而"关"或"静止"状态用 $x_i=-1$ 表示。因此对由 N 个神经元构成的网络来说，网络状态由向量 $\mathbf{x}=[x_1,x_2,\cdots,x_N]^T$ 定义。由于 $x_i=\pm 1$，神经元 i 的状态表示1比特信息，而 $N\times 1$ 的向量 $\mathbf{x}$ 表示 N 比特信息的二进制字。

神经元 j 的诱导局部域 v_j 定义为

$$v_j = \sum_{i=1}^{N} w_{ji}x_i + b_j \tag{14.40}$$

其中 b_j 是额外施加在神经元 j 上的固定偏置。因此，神经元 j 根据确定性规则

$$x_j = \begin{cases} +1, v_j > 0 \\ -1, v_j < 0 \end{cases}$$

修改它的状态 x_j。

这一关系可以改写为紧凑形式 $x_j=\mathrm{sgn}[v_j]$，其中sgn是符号函数。如果 v_j 恰好是零会出现什么情况？在这里采取的行动可能是非常任意的。例如，如果 $v_j=0$，我们可以设置 $x_j=\pm 1$。然而，我们将使用如下约定：如果 v_j 是零，神经元 j 保持它原有状态，不管它是开还是关。就像将在后面说明的那样，这一假定的显著意义在于作为结果的流图表是对称的。

把离散Hopfield网络作为按内容寻址存储器的操作有两个阶段，即存储阶段和检索阶段，如下面说明的那样。

1. *存储阶段*。假设我们希望存储一组表示为 $\{\xi_\mu|\mu=1,2,\cdots,M\}$ 的 N 维向量(二进制字)集合。我们称这 M 个向量为*基本记忆*，表示被网络存储的模式。令 $\xi_{\mu,i}$ 表示基本记忆 ξ_μ 的第 i 个元素，其中类 $\mu=1,2,\cdots,M$。根据存储的*外积规则*，也就是*Hebb学习的基本原则的*推广，从神经元 i 到神经元 j 的突触权值定义为

$$w_{ji} = \frac{1}{N}\sum_{\mu=1}^{M}\xi_{\mu,j}\xi_{\mu,i} \tag{14.41}$$

688

使用 1/N 作为比例常数的原因是为了简化信息检索的数学表述。也要注意式(14.41)的学习规则是"单射"(one shot)计算。在 Hopfield 网络正常运行中，我们设置

$$w_{ii} = 0 \qquad 对于所有的\ i \tag{14.42}$$

这意味着神经元没有自反馈。令 **W** 表示网络 $N \times N$ 的突触权值矩阵，用 w_{ji} 作为它的第 ji 个元素。从而我们可以把式(14.41)和式(14.42)用矩阵形式组合为如下的等式：

$$\mathbf{W} = \frac{1}{N}\sum_{\mu=1}^{M}\xi_{\mu}\xi_{\mu}^{T} - M\mathbf{I} \tag{14.43}$$

其中 $\xi_{\mu}\xi_{\mu}^{T}$ 表示向量 ξ_{μ} 和它自身的外积，而 **I** 表示单位矩阵。从这一突触权值集/权值矩阵的定义式我们可以重新确认如下事实：

- 网络中每一神经元的输出都反馈到所有的其他神经元上。
- 网络中没有自反馈(即 $w_{ii}=0$)。
- 网络权值矩阵是对称的，表示为(参照式(14.21))

$$\mathbf{W}^{T} = \mathbf{W} \tag{14.44}$$

2. 检索阶段。在检索阶段，一个称为探针(probe)的 N 维向量 ξ_{probe} 被强加于 Hopfield 网络作为它的状态。探针向量的元素为 ±1。它典型地表征网络中基本记忆的不完整或噪声形式。然后信息检索依照动态规则进行，在该规则中网络的每一神经元 j 随机地但按某一固定比率检测作用在其上的诱导局部域 v_j(包含任意非零偏置 b_j)。如果在某一时刻 v_j 大于零，则神经元 j 将切换它的状态到 +1，或者保持在该状态，如果已经是 +1 的话。类似地，如果 v_j 小于零，则神经元 j 将切换它的状态到 −1，或者保持在该状态，如果已经是 −1 的话。如果 v_j 恰好为零，则不管是开还是关，神经元 j 都将保持原有状态。因此，从一个迭代到另一个迭代的状态更新是确定的，但是选择进行更新操作的神经元则是随机的。这里描述的异步(串行)更新过程继续直到没有任何进一步的变化可以报告为止。那就是说，用探针向量 **x** 开始，最终网络生成一个不随时间改变的状态向量 **y**，它的每个元素都满足稳定性条件

$$y_j = \mathrm{sgn}\left(\sum_{i=1}^{N} w_{ji}y_i + b_j\right), j = 1,2,\cdots,N \tag{14.45}$$

689

或者其矩阵形式

$$\mathbf{y} = \mathrm{sgn}(\mathbf{W}\mathbf{y} + \mathbf{b}) \tag{14.46}$$

其中 **W** 是网络突触权值矩阵，**b** 是外部施加的偏置向量。这里描述的稳定性条件也称为对齐(alignment)条件。满足条件的状态向量 **y** 称为系统状态空间的稳定状态或固定点。因此我们可以作这样的陈述，当检索操作异步进行时，Hopfield 网络将肯定收敛于一稳定状态[5]。

表 14-2 提出对 Hopfield 网络操作包括存储阶段和检索阶段的步骤的一个小结。

表 14-2 Hopfield 模型小结

1. 学习。令 $\xi_1, \xi_2, \cdots, \xi_M$ 表示已知 N 维基本记忆的集合。使用外积规则(即 Hebb 学习的基本原则)计算网络的突触权值：

$$w_{ji} = \begin{cases} \frac{1}{N}\sum_{\mu=1}^{M}\xi_{\mu,j}\xi_{\mu,i}, & j \neq i, \\ 0, & j = i \end{cases}$$

（续）

其中 w_{ji} 为从神经元 i 到神经元 j 的突触权值。向量 ξ_μ 的元素等于 ±1。一旦它们被计算出，则突触权值保持不变。

2. 初始化。令 ξ_{probe} 表示出现在网络中的一未知 N 维输入向量（探针）。通过设置

$$x_j(0) = \xi_{j,probe}, j = 1, \cdots, N$$

初始化算法，其中 $x_j(0)$ 是神经元 j 在时间 $n=0$ 时的状态，$\xi_{j,probe}$ 是探针向量 ξ_{probe} 的第 j 个元素。

3. 迭代直到收敛。根据如下规则异步地（即随机并且每次一个地）更新状态向量 $\mathbf{x}(n)$ 中的元素：

$$x_j(n+1) = \text{sgn}\left[\sum_{i=1}^{N} w_{ji} x_i(n)\right], j = 1, 2, \cdots, N$$

重复这一迭代直到状态向量 $\mathbf{x}$ 保持不变。

4. 输出。令 $\mathbf{x}_{fixed}$ 表示第3步计算出的固定点（稳定状态），作为结果的网络输出向量 $\mathbf{y}$ 为

$$\mathbf{y} = \mathbf{x}_{fixed}$$

第1步是存储阶段，第2步到第4步构成检索阶段。

例14.2 为了说明Hopfield模型的再现行为，考虑图14-14a所示的三个神经元的网络。网络权值矩阵为

$$\mathbf{W} = \frac{1}{3}\begin{bmatrix} 0 & -2 & +2 \\ -2 & 0 & -2 \\ +2 & -2 & 0 \end{bmatrix}$$

690

因为它满足式(14.42)和式(14.44)的条件，所以是合法的。假定施加在每个神经元上的偏置为零。由于网络中有三个神经元，所以要考虑的可能状态有 $2^3=8$ 种。这8种状态中，只有(1, -1, 1)和(-1, 1, -1)这两种状态是稳定的；其余的6种状态都是不稳定的。我们说这两种特殊状态是稳定的是因为它们都满足式(14.46)的对齐条件。对状态向量(1, -1, 1)，我们有

$$\mathbf{Wy} = \frac{1}{3}\begin{bmatrix} 0 & -2 & +2 \\ -2 & 0 & -2 \\ +2 & -2 & 0 \end{bmatrix}\begin{bmatrix} +1 \\ -1 \\ +1 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} +4 \\ -4 \\ +4 \end{bmatrix}$$

硬限制这一结果得到

$$\text{sgn}[\mathbf{Wy}] = \begin{bmatrix} +1 \\ -1 \\ +1 \end{bmatrix} = \mathbf{y}$$

类似地，对状态向量(-1, 1, -1)，我们有

$$\mathbf{Wy} = \frac{1}{3}\begin{bmatrix} 0 & -2 & +2 \\ -2 & 0 & -2 \\ +2 & -2 & 0 \end{bmatrix}\begin{bmatrix} -1 \\ +1 \\ -1 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} -4 \\ +4 \\ -4 \end{bmatrix}$$

硬限制这一结果之后，得到

$$\text{sgn}[\mathbf{Wy}] = \begin{bmatrix} -1 \\ +1 \\ -1 \end{bmatrix} = \mathbf{y}$$

因此，这两种状态向量都满足对齐条件。

此外，遵从表14-2小结的异步更新过程，我们得到图14-14b所描绘的流。这个流图展

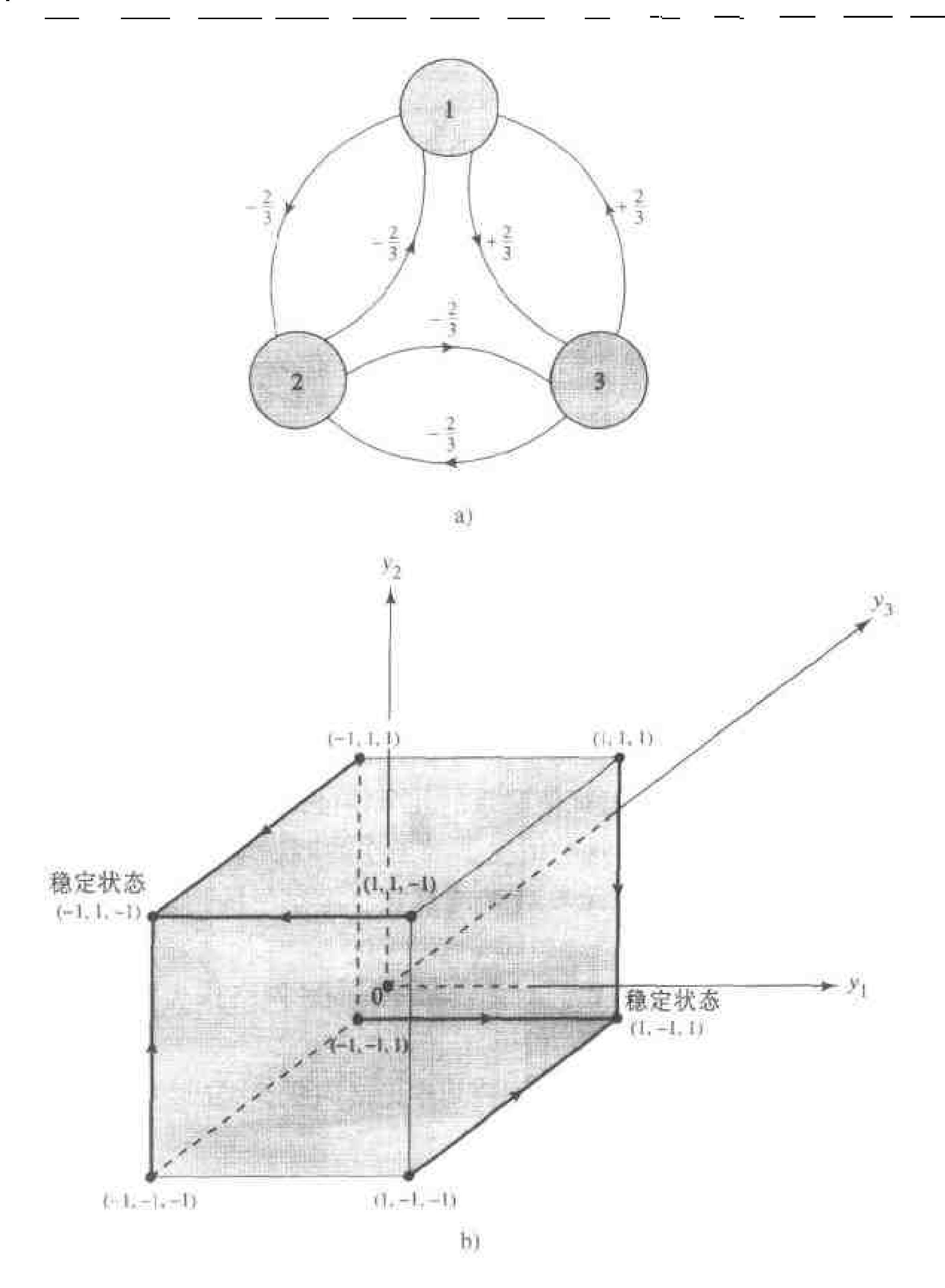


图 14-14

a) $N=3$ 个神经元的 Hopfield 网络结构图 b) 描绘两个稳定态和网络流的图

示关于网络中直观上满足条件的两个稳定状态之间的对称性。这种对称性是令作用于其上的诱导局部域恰好为零的神经元保留在原有状态的结果。 691

图 14-14b 也显示出如果图 14-14a 的网络初始状态是(1,1,1)、(-1,-1,1)或(1,-1,-1)，那么在一次迭代之后它将收敛于稳定状态(1,-1,1)。如果初始状态是(-1,-1,-1)、(-1,1,1)或(1,1,-1)，则它将收敛于第二个稳定状态(-1,1,-1)。

因此，网络有两个基本记忆(1,-1,1)和(-1,1,-1)表征这两个稳定状态。式(14.43)的应用产生突触权值矩阵

$$\mathbf{W}=\frac{1}{3}\begin{bmatrix}+1\\-1\\+1\end{bmatrix}[+1,-1,+1]+\frac{1}{3}\begin{bmatrix}-1\\+1\\-1\end{bmatrix}[-1,+1,-1]-\frac{2}{3}\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

$$= \frac{1}{3}\begin{bmatrix} 0 & -2 & +2 \\ -2 & 0 & -2 \\ +2 & -2 & 0 \end{bmatrix}$$

它和图14-14a所示的突触权值符合。

通过检验图14-14b的流图，Hopfield网络的纠错能力是显而易见的：

1. 如果作用在网络上的探针向量 ξ_{probe} 等于 $(-1,-1,1)$、$(1,1,1)$ 或 $(1,-1,-1)$，则作为结果的输出是基本记忆 $(1,-1,1)$。每个这样的探针的值表示一个和存储模式相比的单一错误。

2. 如果探针向量 ξ_{probe} 等于 $(1,1,-1)$、$(-1,-1,-1)$ 或 $(-1,1,1)$，则作为结果的输出是基本记忆 $(-1,1,-1)$。这里再次表明，每个这样的探针表示一个和存储模式相比的单一错误。 ■

伪状态

就像式(14.44)指出的那样，离散Hopfield网络的权值矩阵 **W** 是对称的。因此 **W** 的特征值都是实数。然而，当 M 很大的时候特征值通常是*退化的*(degenerate)，这意味着有几个特征向量有同样的特征值。通过退化特征值联系的几个特征向量构成了一个子空间。此外，权值矩阵 **W** 退化特征值有等于零的，这种情况下的子空间叫做*零空间*。零空间的存在是由于基本记忆的数量 M 小于网络中神经元数量 N 的事实。零空间的出现是Hopfield网络的内在特性。[692]

权值矩阵 **W** 的特征分析，使得我们对把离散Hopfield网络作为按内容寻址存储器持下列观点(Aiyer et al.,1990)：

1. 离散Hopfield网络将探针向量投影到被基本记忆向量扩张成的子空间 $\mathcal{M}$ 上，从这种意义上说，它起到*向量投影器*的作用。

2. 网络固有的动力学把结果投影向量驱动到单位超立方体的能量函数最小的一个角点处。

单位超立方体是 N 维的。扩张成子空间 $\mathcal{M}$ 的 M 个基本记忆向量组成由单位超立方体确定的角点表示的固定点(稳定状态)的集合。单位超立方体的其他位于子空间 $\mathcal{M}$ 内部或附近的角点是潜在*伪状态*(spurious states)的所在位置，也称为*伪吸引子*(Amit,1989)。伪状态表示Hopfield网络中不同于网络基本记忆的其他稳定状态。

因此，在设计作为按内容寻址存储器的Hopfield网络过程中，我们面临着对两个矛盾需求的权衡：(1)需要在状态空间中保持基本记忆向量作为固定点，(2)希望有少量的伪状态。

Hopfield网络的存储容量

不幸的是，Hopfield网络的基本记忆不总是稳定的。而且，可能出现由伪状态表征的不同于基本记忆的其他稳定状态。这两个现象倾向于降低作为按内容寻址存储器的Hopfield网络的效率。在这里我们探索一下第一个现象。

令探针等于作用于网络上的基本记忆中的一个 ξ_ν。然后，为了一般性允许使用自反馈并设定零偏置，我们发现使用式(14.41)，则神经元 j 的诱导局部域为：

$$v_j = \sum_{i=1}^{N} w_{ji}\xi_{\nu,i} = \frac{1}{N}\sum_{\mu=1}^{M}\xi_{\mu,j}\sum_{i=1}^{N}\xi_{\mu,i}\xi_{\nu,i} = \xi_{\nu,j} + \frac{1}{N}\sum_{\substack{\mu=1\\ \mu\neq\nu}}^{N}\xi_{\mu,j}\sum_{i=1}^{N}\xi_{\mu,i}\xi_{\nu,i} \qquad (14.47)$$

式(14.47)右端第一项只是基本记忆 ξ_ν 的第 j 个元素；现在我们可以看出比例因子 $1/N$ 为什么被引入式(14.41)中突触权值 w_{ji} 的定义中。因此这一项可以被看作 v_j 期望中的"信号"成分。式(14.47)右端第二项是在被测基本记忆 ξ_ν 的元素和其他基本记忆 ξ_μ 的元素之间的"串音"(crosstalk)的结果。因而这第二项可以被看作 v_j 的"噪声"成分。因此我们有了和通信理论中典型的"带噪声信号检测问题"类似的情景(Haykin, 1994b)。 [693]

我们假设基本记忆是随机的和作为 MN 个 Bernoulli 实验序列生成的。那么式(14.47)中的噪声项构成 $N(M-1)$ 个取值为 ± 1 的独立随机变量的求和除以 N。这正是使用概率论中的中心极限定理的情形。中心极限定理陈述如下(Feller, 1968)：

令 $\{X_k\}$ 为同分布的互相独立随机变量序列。假设 X_k 具有均值 μ 和方差 σ^2，令 $Y = X_1 + X_2 + \cdots + X_n$。那么当 n 趋向无限时，求和随机变量 Y 的概率分布趋于 Gauss 分布。

因此，通过在式(14.47)中噪声项上采用中心极限定理，我们发现噪声是渐近的 Gauss 分布。构成等式中噪声项的这 $N(M-1)$ 个随机变量中的每一个都有均值 0 和方差 $1/N^2$。因而，推知高斯分布的统计学性质为

- 均值零
- 方差等于 $(M-1)/N$

信号成分 $\xi_{\nu,j}$ 等于值 +1 或 -1 的概率相等，并因此有均值 0 和方差 1。所以信噪比(signal-to-noise ratio)定义为

$$\rho = \frac{\text{信号方差}}{\text{噪音方差}} = \frac{1}{(M-1)/N} \simeq \frac{N}{M} \qquad \text{对于很大的 } M \qquad (14.48)$$

基本记忆 ξ_ν 的成分当且仅当信噪比 ρ 高的时候才是稳定的。现在，基本记忆的数量 M 提供直接度量网络存储容量(storage capacity)的方法。因此，只要网络存储容量不超载，也就是说基本记忆数量 M 比网络中神经元数量 N 要小，由式(14.48)可得基本记忆从概率意义上是稳定的。

信噪比的倒数，也就是

$$\alpha = \frac{M}{N} \qquad (14.49)$$

[694]

称为负载参数(load parameter)。统计物理学的考虑显示出 Hopfield 网络的记忆检索的质量随负载参数 α 的增加而恶化，并且在临界值 $\alpha_c = 0.14$ 处崩溃(Amit, 1989; Müller and Reinhardt, 1990)。这一临界值与 Hopfield(1982)的估计相符，其中作为计算机模拟的结果报告 0.15 N 个状态可以在错误变得严重之前同时被检索出。

由于 $\alpha_c = 0.14$，我们从式(14.48)发现信噪比的临界值 $\rho_c \simeq 7$，或者等价的 8.45 分贝。至于信噪比低于这一临界值，则记忆检索崩溃。

临界值

$$M_c = \alpha_c N = 0.14N \qquad (14.50)$$

定义检索的容错存储容量。为了确定不带错误的存储容量，我们必须使用下面描述的错误概率定义的更严格准则。

令探针 $\xi_{\text{probe}} = \xi_\nu$ 的第 j 位为符号 1，也就是 $\xi_{\nu,j} = 1$。那么检索时第 j 位出错的条件概率

由图 14-15 中的阴影区域定义。这一曲线下的其余区域为探针第 j 位正确恢复的条件概率。使用熟知的高斯分布公式，后一条件概率由下式给出：

$$P(v_j > 0 \mid \xi_{\nu,j} = +1) = \frac{1}{\sqrt{2\pi}\sigma}\int_0^{\infty} \exp\left(-\frac{(v_j - \mu)^2}{2\sigma^2}\right) dv_j \tag{14.51}$$

由于 $\xi_{\nu,j}$ 置为 +1，并且式(14.47)中噪声项的均值等于零，由此推出随机变量 V 的均值为 $\mu = 1$，方差为 $\sigma^2 = (M-1)/N$。从通常用于涉及高斯分布的计算的误差函数定义，我们有

$$\mathrm{erf}(y) = \frac{2}{\sqrt{\pi}}\int_0^{y} e^{-z^2} dz \tag{14.52}$$

其中 y 为定义积分上限的变量。现在通过误差函数把式(14.51)改写成

$$P(v_j > 0 \mid \xi_{\nu,j} = +1) = \frac{1}{2}\left[1 + \mathrm{erf}\left(\sqrt{\frac{\rho}{2}}\right)\right] \tag{14.53}$$

其中 ρ 是式(14.48)定义的信噪比，我们可以简化正确恢复基本记忆 ξ_ν 第 j 位的条件概率的表达式。每个基本记忆包含 n 位。同时，基本记忆通常是等概率的。因此稳定模式的概率定义为

695

$$P_{\mathrm{stab}} = (P(v_j > 0 \mid \xi_{\nu,j} = +1))^N \tag{14.54}$$

我们可以使用这一概率来构成 Hopfield 网络容量的表达式。具体地，我们定义几乎不带错误的存储容量 $M_{\max}$ 作为网络中能够存储的最大基本记忆数量，并且强调它们中的绝大部分能被正确检索。在习题 14.8 中证明由这个存储容量的定义得到公式

$$M_{\max} = \frac{N}{2\log_e N} \tag{14.55}$$

其中 $\log_e$ 表示自然对数。

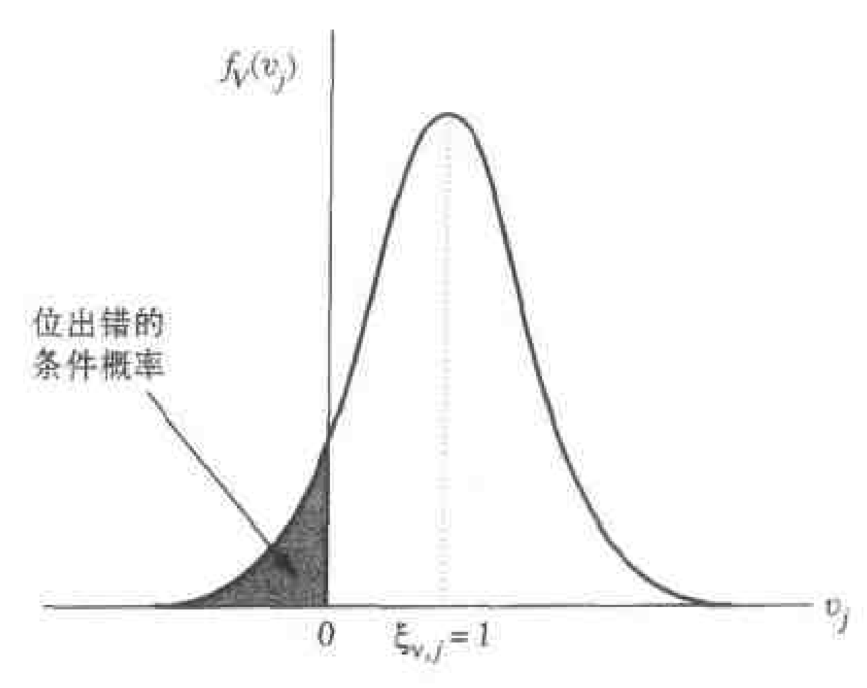


图 14-15 位出错的条件概率，假设神经元 j 的诱导局部域 v_j 为高斯分布概率密度函数 $f_V(v_j)$ 的下标 V 表示随机变量，v_j 表示它的实现

图 14-16 画出式(14.50)定义的带错误存储容量和式(14.55)定义的几乎不带错误存储容量两者对于网络大小 N 的关系图形。从该图中我们注意以下两点：

- Hopfield 网络的存储容量本质上与网络大小 N 成线性关系。
- Hopfield 网络的主要局限在于，为了基本记忆的可恢复性，它的存储容量必须维持很小[6]。

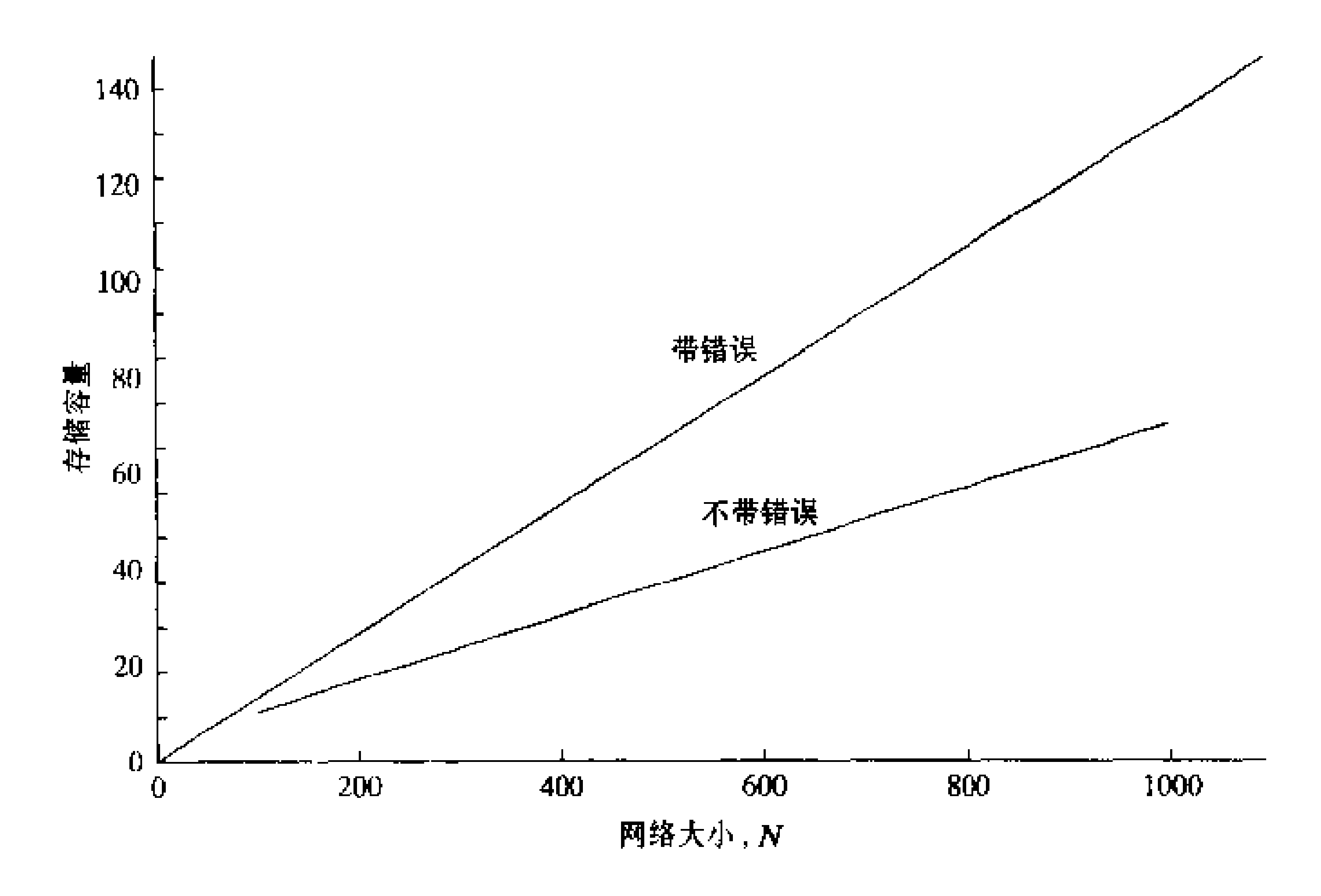


图 14-16 Hopfield 网络规模和它的两种网络容量的关系图：带错误和几乎不带错误的

14.8 计算机实验 I

这一节中我们用计算机实验来说明作为按内容寻址存储的离散 Hopfield 网络的性能。实验中使用的网络包含 $N=120$ 个神经元，因此有 $N^2-N=12\ 280$ 个突触权值。它被训练用于恢复图 14-17 中的 8 个数字的黑白图样，每个图样有 120 个象素(图元素)并特别设计以产生良好的性能(Lippmann,1987)。在网络的输入中设定用值 +1 表示黑象素，-1 表示白象素。在 Hopfield 网络的存储(学习)阶段，图 14-17 中的 8 个图样被用作基本记忆使用式(14.43)生成突触权值矩阵 **W**。网络操作的检索阶段像表 14-2 中说明的那样异步进行。 [696]

在实验恢复部分的第一阶段，基本记忆被提交给网络，检验从突触权值矩阵存储的信息中正确恢复它们的能力。每一种情况下，希望得到的图样都在一次迭代之后由网络生成了。

下一步，为了验证 Hopfield 网络的纠错能力，通过使用 0.25 的概率随机地和独立地从 +1 到 -1反转每一个象素，并反过来进行，这样随机扭曲一个感兴趣的图样，然后使用这个被破坏的图样作为网络的探针。对数字 3 的实验结果如图 14-18 所示。图中上部分表示数字 3 的被破坏版本，也就是在时刻零作用在网络上的图样。网络在 5 次、10 次、15 次、20 次、25 次、30 次和 35 次迭代之后生成的图样在图中其余部分给出。随着迭代次数的增加，我们看到网络输出和数字 3 的类同之处逐步提高。事实上，在 35 次迭代之后网络已收敛在数字 3 的准确形式。 [697]

理论上对每个被破坏图样因为 hopfield 网络中有 120 个神经元的四分之一改变状态，所以检索所需迭代数量平均值为 30。在我们的实验中，对不同图样从它们被破坏形式进行检索所需迭代数量如下：

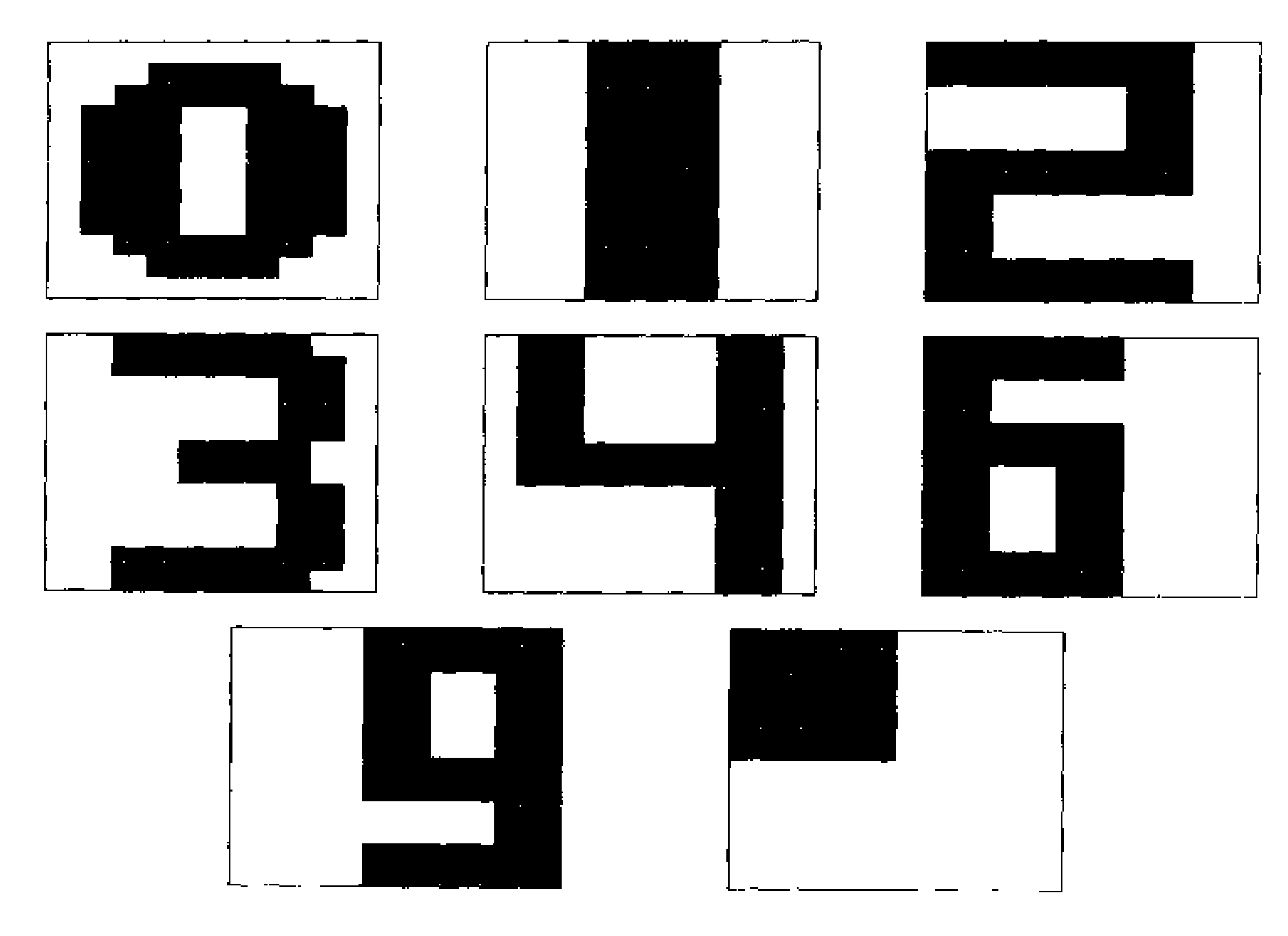


图 14-17 用于 Hopfield 网络计算机实验的(人工)图样集

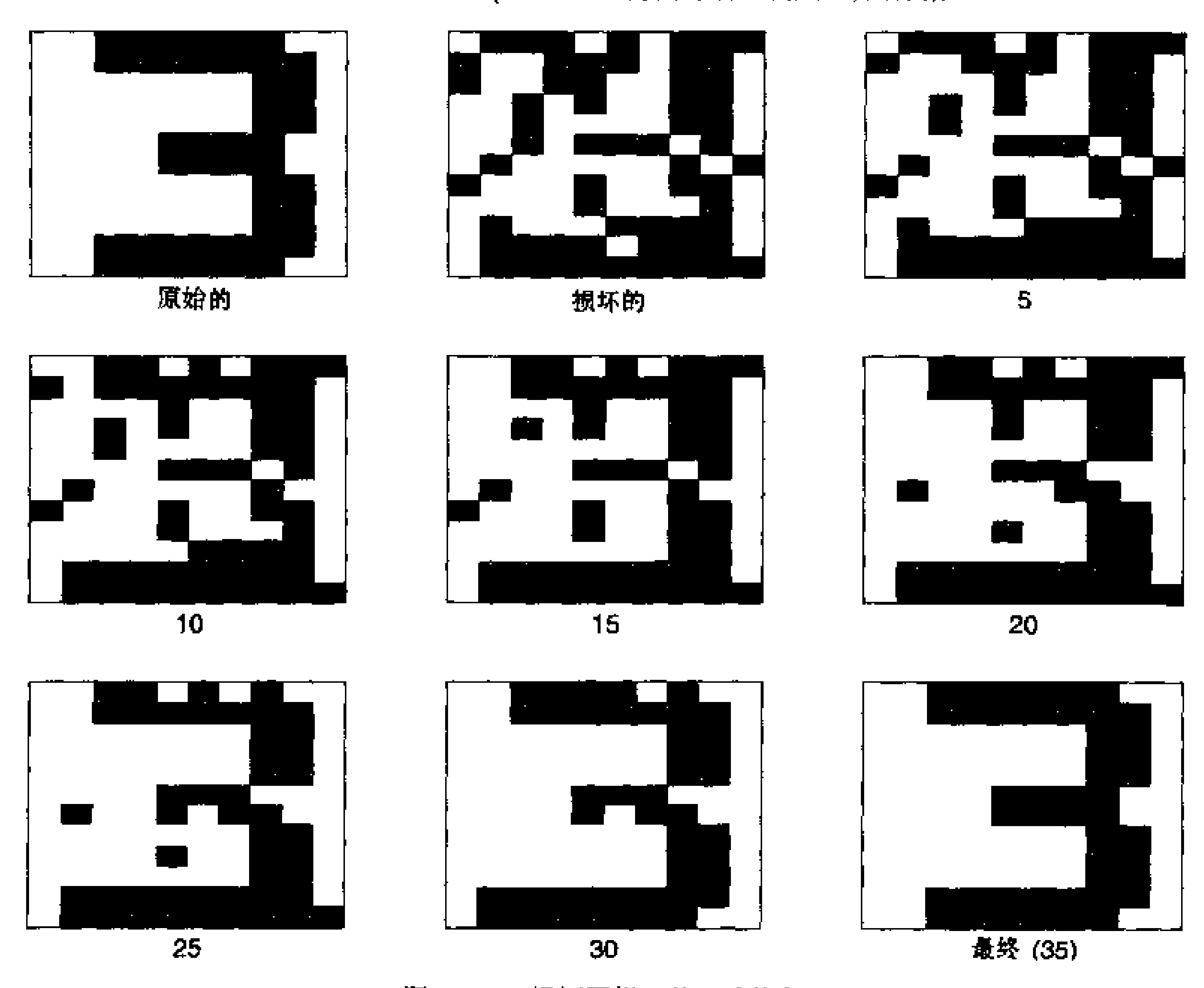


图 14-18 损坏图样 3 的正确恢复

图样	检索所需图样数量
0	34
1	32
2	26
3	35
4	25
6	37
"■□"	32
9	26

检索所需迭代次数在 8 个图样上平均所得平均值大约是 31，这表明 Hopfield 网络像预期的那样运转。 698

Hopfield 网络固有的问题出现在一个基本记忆的被破坏版本提交给网络的时候，然后随着网络的运行收敛在一个错误的基本记忆上。这一切在图 14-19 中说明：其中提交给网络的是被破坏图样"2"，但是在 47 次迭代之后网络收敛在基本记忆"6"上了。

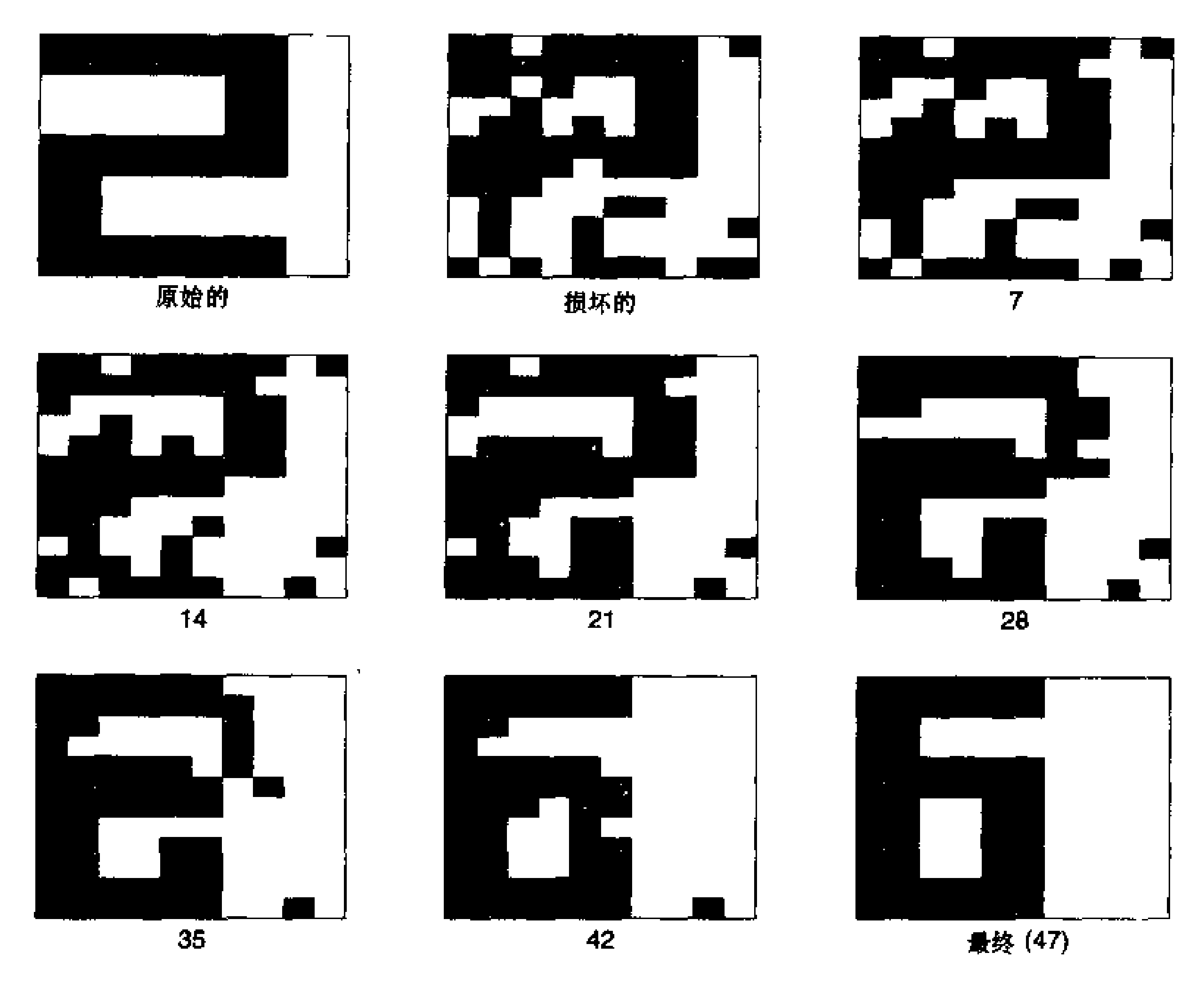


图 14-19 损坏图样 2 的错误恢复

就像前面提到的那样，在 Hopfield 网络中还出现另外一个问题：伪状态的存在。图 14-20(视为 14×8 的网络状态矩阵)给出在 43 097 次对随机选择的数字按 0.25 的概率翻转 1 位被破坏的检验中发现的 108 种伪吸引子。伪状态可以分组如下(Amit,1989)：

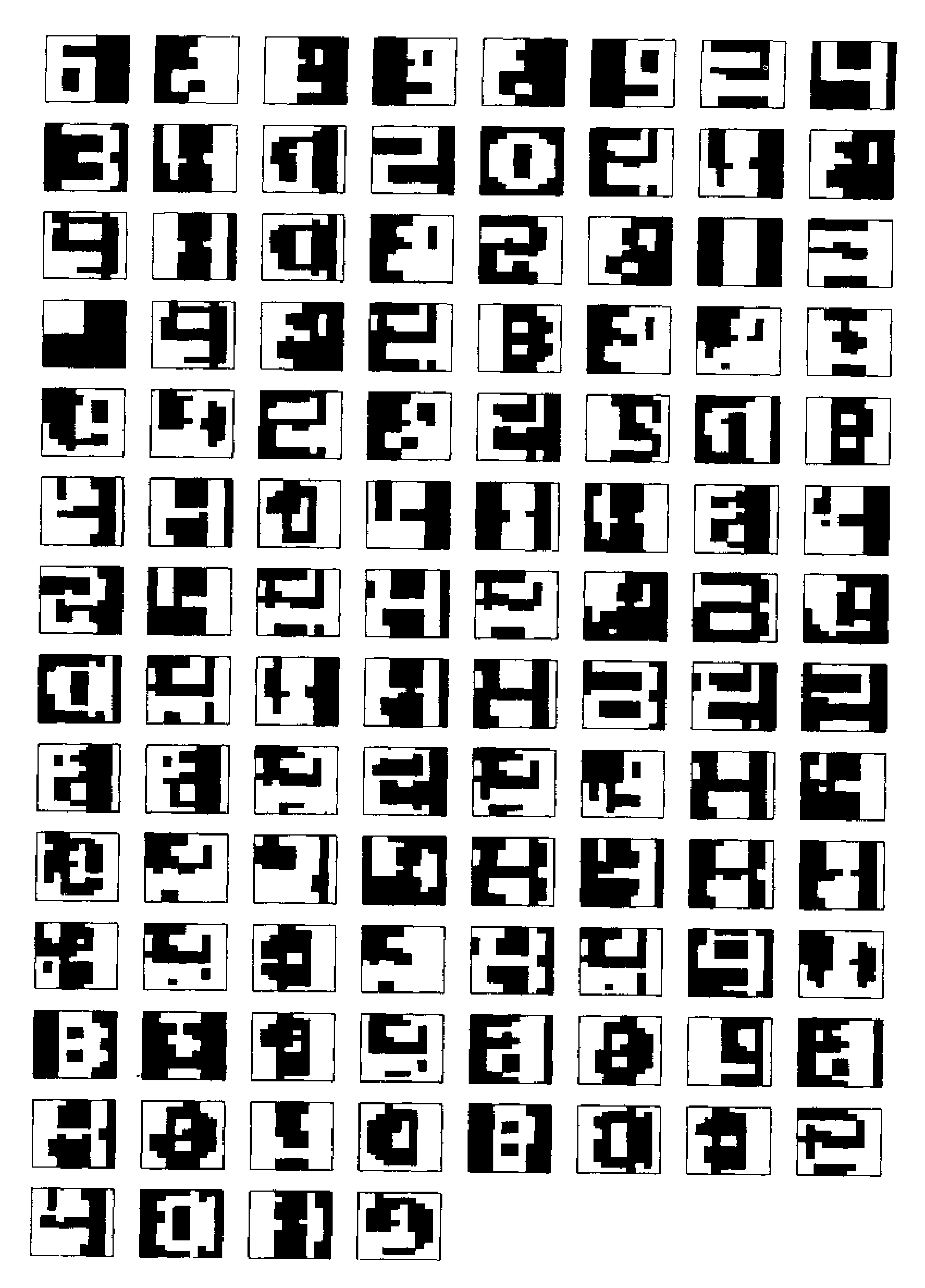


图 14-20 Hopfield 网络计算机实验产生的伪状态汇集

1. *反基本记忆*。这些伪状态是网络基本记忆的反转(即负的)版本；例如，参考图 14-20 中位置 1×1 处的状态，它表示图 14-17 中数字 6 的负数。为了解释这类伪状态，我们注意能量函数 E 在神经元状态反转的时候保持它的值不变，从这种意义上说它是对称的(即对于所有的 i，状态 x_i 用 $-x_i$ 替换)。因此，如果基本记忆 ξ_μ对应能量等值线的某一特定局部极小值，同样该最小值也对应 $-\xi_\mu$。如果被恢复模式的所有信息位都被反转的话，也就是如果可以发现那些特定的位即设计为 -1 的“符号”位被 +1 替换的话，则这一符号反转不会给信息恢复带来问题。 699

2. *混合状态*。混合(mixture)伪状态是奇数个被存储模式的线性组合。例如，考虑状态

$$x_i = \text{sgn}(\xi_{1,i} + \xi_{2,i} + \xi_{3,i})$$

这是一个三混合伪状态。它由三个基本记忆 ξ_1、ξ_2 和 ξ_3 通过多数原则形成的。对大型网络，这样的状态是满足式(14.45)的稳定条件的。图 14-20 中第 6 行第 4 列位置的图样代表一个由以下基本记忆组成的三混合伪状态：ξ_1 = 负的数字 1，ξ_2 = 数字 4，ξ_3 = 数字 9。

3. *旋转玻璃状态*。这种伪状态这样命名与统计力学的旋转玻璃模型类似。旋转玻璃状态由没有和网络中基本记忆相互关联的能量等值线的局部最小值定义；例如，参看图 14-20 中第 7 行第 6 列处的状态。 700

14.9 Cohen-Grossberg 定理

在 Cohen-Grossberg(1983)，给出评价由如下联立非线性微分方程组描述的一类神经网络的稳定性的一般原则：

$$\frac{d}{dt}u_j = a_j(u_j)\left[b_j(u_j) - \sum_{i=1}^{N} c_{ji}\varphi_i(u_i)\right], j = 1,\cdots,N \tag{14.56}$$

根据 Cohen-Grossberg 定理，这类神经网络容许定义一个 Lyapunov 函数(看习题 14.13)

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ji}\varphi_i(u_i)\varphi_j(u_j) - \sum_{j=1}^{N}\int_0^{u_j} b_j(\lambda)\varphi'_j(\lambda)\,d\lambda \tag{14.57}$$

其中

$$\varphi'_j(\lambda) = \frac{d}{d\lambda}(\varphi_j(\lambda)) \tag{14.58}$$

为了使式(14.57)的定义有效，需要下面条件成立：

1. 网络的突触权值对称：

$$c_{ij} = c_{ji} \tag{14.59}$$

2. $a_j(u_j)$满足非负性条件：

$$a_j(u_j) \geqslant 0 \tag{14.60}$$

701

3. 非线性输入 - 输出函数满足单调性条件：

$$\varphi'_j(u_j) = \frac{d}{du_j}\varphi_j(u_j) \geqslant 0 \tag{14.61}$$

现在，我们可以正式地陈述 Cohen-Grossberg 定理：

如果非线性微分方程组(14.56)满足对称性、非负性和单调性，则由式(14.57)描述的 Lyapunov 函数 E 满足条件

$$\frac{dE}{dt} \leqslant 0$$

一旦 Lyapunov 函数 E 的基本属性具备，系统的全局稳定性从 Lyapunov 定理 1 推出。

Hopfield 模型作为 Cohen-Grossberg 定理的特例

对一个连续的 Hopfield 模型，通过比较方程组(14.56)和方程组(14.20)，我们可以得到 Hopfield 模型和 Cohen-Grossberg 定理之间的对应关系，这种关系如表 14-3 所示。在式(14.57)中运用此表，就可以得到连续的 Hopfield 模型的 Lyapunov 函数

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji}\varphi_i(v_i)\varphi_j(v_j) + \sum_{j=1}^{N}\int_0^{v_j}\left(\frac{v_j}{R_j} - I_j\right)\varphi'_j(v)\,dv \tag{14.62}$$

其中非线性激活函数 $\varphi_j(\cdot)$ 由式(14.23)定义。

接下来，我们得到如下的观察结果：

1. $\varphi_i(v_i) = x_i$

2. $\int_0^{v_j}\varphi'_j(v)\,dv = \int_0^{x_j} dx = x_j$

3. $\int_0^{v_j} v\varphi'_j(v)\,dv = \int_0^{x_j} dx = \int_0^{x_j}\varphi_j^{-1}(x)\,dx$

从基本上说，关系式 2 和 3 通过应用 $x = \varphi_i(v)$ 得到。这样，在式(14.62)的 Lyapunov 函数中运用这些观察就可以得到和我们早先描述的相同的结果；参看式(14.28)。然而，尽管 $\varphi_i(v)$ 必须是输入 v 的非减函数，为使式(14.62)描述的通用 Lyapunov 函数成立，并不需要 $\varphi_j(v)$ 是可逆的。

Cohen-Grossberg 定理是有广泛应用的神经动力学的一个基本原理。在下一节我们考虑这个重要定理的另一个应用。

表 14-3 Cohen-Grossberg 定理和 Hopfield 模型的对应关系

Cohen-Grossberg 定理	Hopfield 模型
u_j	$C_j v_j$
$a_j(u_j)$	1
$b_j(u_j)$	$-(v_j/R_j) + I_j$
c_{ji}	$-w_{ji}$
$\varphi_i(u_i)$	$\varphi_i(v_i)$

702

14.10 盒中脑状态模型

在这一节中，我们通过学习盒中脑状态(brain-state-in-a-box, BSB)模型来继续联想记忆的神经动力学的分析。该模型首先由 Anderson et al.(1977)描述。BSB 模型基本上是一个带幅度限制的正反馈系统，该模型是由一组反馈回自身的高度互连的神经元组成。模型用内置的正反馈来放大输入模式，直到模型中的所有神经元饱和。这样，BSB 模型可以看作一个分类器，在该分类器中，给定一个模拟输入模式，产生一个由模型稳定状态描述的数字表示。

用 $\mathbf{W}$ 表示对称权值矩阵，该矩阵的最大特征值为正实数。用 $\mathbf{x}(0)$ 表示模型的初始状态向量，代表输入激活模式。假定模型中有 N 个神经元。模型的状态向量是 N 维的，$\mathbf{W}$ 是 $N\times N$ 矩阵。BSB 算法由下面两个方程完全定义：

$$\mathbf{y}(n) = \mathbf{x}(n) + \beta\mathbf{W}\mathbf{x}(n) \tag{14.63}$$

$$\mathbf{x}(n+1) = \varphi(\mathbf{y}(n)) \tag{14.64}$$

其中β是一个称为反馈因子的正的小常数，$\mathbf{x}(n)$是模型在时刻 n 的状态向量。图 14-21a 显示式(14.63)和式(14.64)的联合框图。方框 $\mathbf{W}$ 代表一个单层线性神经网络，如图 14-21b 所示。激活函数 φ 是一个作用在 $y_j(n)$上的分段线性函数，$y_j(n)$是向量 $\mathbf{y}(n)$的第 j 个分量，如下所示(看图 14-22)

$$x_j(n+1) = \varphi(y_j(n)) = \begin{cases} +1 & \text{当 } y_j(n) > +1 \\ y_j(n) & \text{当 } -1 \leqslant y_j(n) \leqslant +1 \\ -1 & \text{当 } y_j(n) < -1 \end{cases} \tag{14.65}$$

703

式(14.65)限制 BSB 模型的状态向量处于中心在原点的一个 N 维单位立方体中。

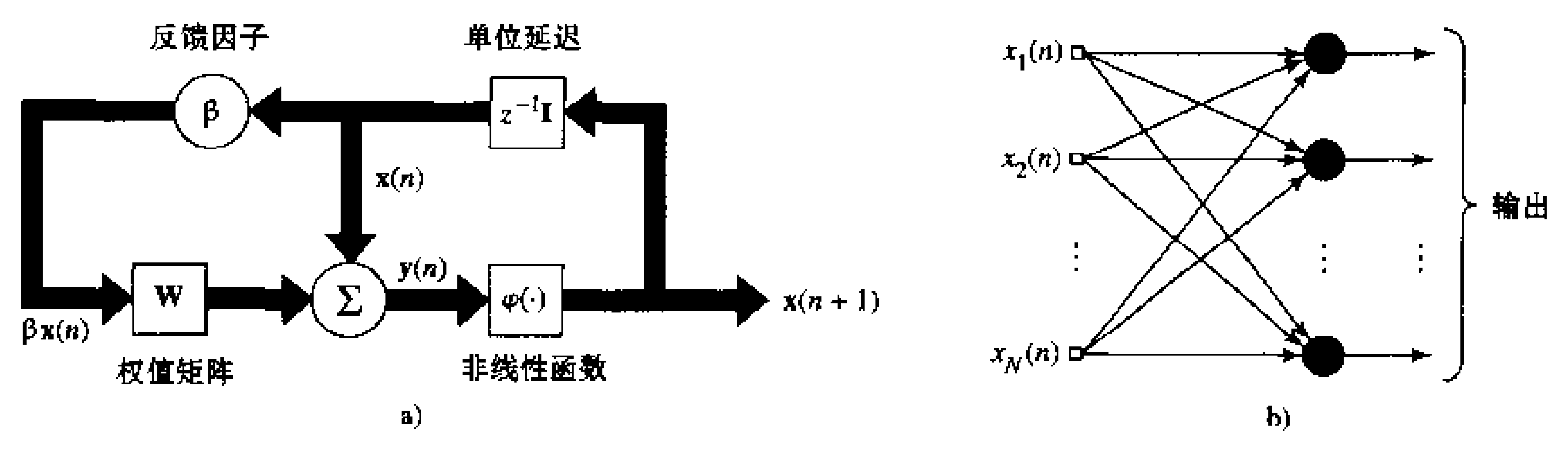


图 14-21
a)盒中脑状态(BSB)模型框图 b)权值矩阵 W 表示的线性联想器的信号流图

算法如下进行：一个激活模式 $\mathbf{x}(0)$ 作为一个初始状态向量输入 BSB 模型，式(14.63)用来计算向量 $\mathbf{y}(0)$，式(14.64)用来截断 $\mathbf{y}(0)$，获得更新状态向量 $\mathbf{x}(1)$。接着，$\mathbf{x}(1)$通过(14.63)和(14.64)循环得到 $\mathbf{x}(2)$。这个过程一直重复直到 BSB 模型达到一个稳定状态，该状态代表超立方体的一个角点。直觉上，BSB 模型的正反馈引起初始状态向量 $\mathbf{x}(0)$的 Euclid 长度(范数)随迭代次数的增加而增加，直到它撞到盒子(单位超立方体)的墙上，然后顺着墙滑行，最终停在盒子的一个稳定角点上，在这里它继续“推进”却不能脱离盒子(Kawamoto and Anderson 1985)，这就是该模型名字的由来。

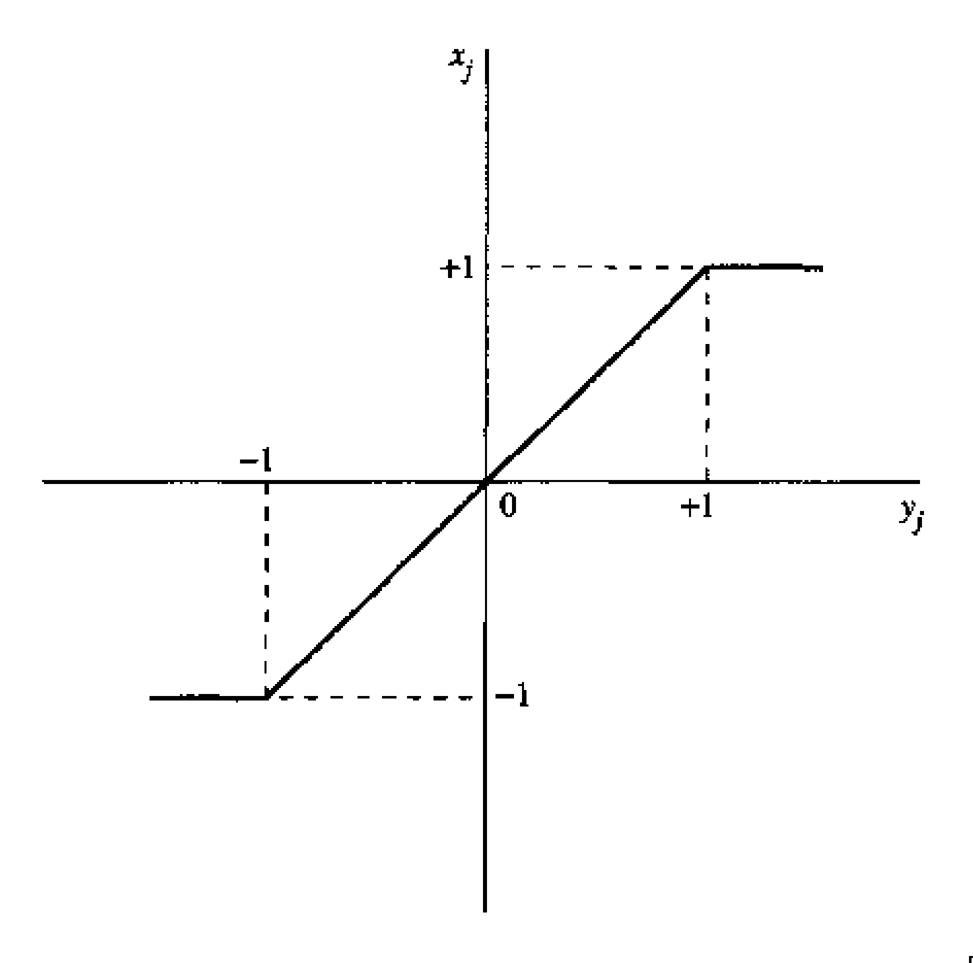


图 14-22 BSB 模型使用的分段线性函数

704

BSB 模型的 Lyapunov 函数

重新定义BSB模型可以作为由式(14.16)描述的神经动力学模型的一个特例(Grossberg, 1990)。为了看到这一点，首先以下述形式重写由式(14.63)和(14.64)描述的 BSB 算法的第 j 个组成部分：

$$x_j(n+1)=\varphi\Big(\sum_{i=1}^{N}c_{ji}x_i(n)\Big),\quad j=1,2,\cdots,N \tag{14.66}$$

系数 c_{ji} 由

$$c_{ji}=\delta_{ji}+\beta w_{ji} \tag{14.67}$$

定义，其中 δ_{ji} 为 Kronecher δ 函数，仅当 $j=i$ 时为 1，其余情况为 0；w_{ji} 是权矩阵 $\mathbf{W}$ 的第 ji 个元素。式(14.66)是离散的时间形式。为了进一步处理，重新用连续时间形式写出它的公式

$$\frac{d}{dt}x_j(t)=-x_j(t)+\varphi\Big(\sum_{i=1}^{N}c_{ji}x_j(t)\Big),j=1,2,\cdots,N \tag{14.68}$$

其中偏置 I_j 对所有的 j 都为 0。然而，为了应用 Cohen-Grossberg 定理，必须进一步把式(14.68)转换成加性模型的形式。我们可以通过引入一组新变量

$$v_j(t)=\sum_{i=1}^{N}c_{ji}x_i(t) \tag{14.69}$$

来做到这点。然后，通过式(14.67)中 c_{ji} 的定义，发现

$$x_j(t)=\sum_{i=1}^{N}c_{ji}v_i(t) \tag{14.70}$$

相应地，重置式(14.68)的模型为等价形式

$$\frac{d}{dt}v_j(t)=-v_j(t)+\sum_{i=1}^{N}c_{ji}\varphi(v_i(t)),j=1,2,\cdots,N \tag{14.71}$$

现在，我们准备把 Cohen-Grossberg 定理应用到 BSB 模型上。通过比较式(14.71)和(14.56)，得到如表 14-4 所示的 BSB 模型和 Cohen-Grossberg 定理的对应关系。因此，把表 14-4 的结果用于式(14.57)，就得到 BSB 模型的 Lyapunov 函数

705

$$E=-\frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}c_{ji}\varphi(v_j)\varphi(v_i)+\sum_{j=1}^{N}\int_0^{v_j}v\varphi'(v)\,dv \tag{14.72}$$

其中 $\varphi'(v)$ 是 sigmoid 函数 $\varphi(v)$ 对它的参数的一阶导数。最后，将式(14.65)，(14.67)和(14.69)的定义代入式(14.72)，就能用原始状态向量定义 BSB 模型的 Lyapunov 函数如下：

$$E=-\frac{\beta}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}w_{ji}x_jx_i=-\frac{\beta}{2}\mathbf{x}^T\mathbf{W}\mathbf{x} \tag{14.73}$$

表 14-4 Cohen-Grossberg 定理和 BSB 模型的对应关系

Cohen-Grossberg 定理	BSB 模型
u_j	v_j
$a_j(u_j)$	1
$b_j(u_j)$	$-v_j$
c_{ji}	$-c_{ji}$
$\varphi_j(u_j)$	$\varphi_j(v_j)$

在 14.7 节中对 Hopfield 网络 Lyapunov 函数的估计，假定模型的非线性 sigmoid 函数的逆的导数存在，此条件是通过用一个双曲线正切函数来满足的。相反，在 BSB 模型中，当第 j 个神经元的状态变量是 +1 或 -1 时，这个条件并不满足。尽管如此，BSB 模型的 Lyapunov 函数能通过 Cohen-Grossberg 定理来估计，从而清楚地表明这个重要定理可以普遍应用。

BSB 模型动力学

在由 Golden(1986)进行的直接分析中，说明 BSB 模型实际是一个梯度下降算法，使得由式(14.73)所定义的能量函数 E 达到最小。然而 BSB 模型的这个重要性质要假设权值矩阵 $\mathbf{W}$ 满足下面两个条件：

- 权值矩阵 $\mathbf{W}$ 是对称的，即 $\mathbf{W}=\mathbf{W}^T$。
- 权值矩阵 $\mathbf{W}$ 是半正定的；也就是说，关于 $\mathbf{W}$ 的特征值，我们有 $\lambda_{min}\geq 0$，其中 λ_{min} 是 $\mathbf{W}$ 的最小特征值。

这样，当在时间 $n+1$ 时的状态向量 $\mathbf{x}(n+1)$ 与在时间 n 的状态向量 $\mathbf{x}(n)$ 不同时，BSB 模型的能量函数 E 随 n(迭代次数)的增加而减小。更进一步，能量函数 E 的最小点定义 BSB 模型的平衡状态，模型由 706

$$\mathbf{x}(n+1)=\mathbf{x}(n)$$

表征。换句话说，像 Hopfield 模型一样，BSB 模型是一个能量最小化网络。

BSB 模型的平衡状态由单位超立方体的特定的角点和它的原点定义。在后一种情况(在原点)，状态向量的任何波动，无论是多么小，都被模型中的正反馈放大，因此引起模型从原点向稳定状态漂移；换句话说，原点是一个鞍点。对超立方体来说，要使它的每个角点作为 BSB 模型的平衡状态，权值矩阵 $\mathbf{W}$ 必须满足第三个条件(Greenberg 1988)：

- 权矩阵 $\mathbf{W}$ 是对角优势的(dominant)，其含义是

$$w_{jj}\geq\sum_{i\neq j}|w_{ij}| \qquad 对所有的\ j=1,2,\cdots,N \tag{14.74}$$

其中 w_{ij} 是 $\mathbf{W}$ 的第 ij 个元素。

为了使平衡状态 $\mathbf{x}$ 稳定，也就是为了使单位超立方体的一个特定角是一个固定点吸引子(attractor)，在单位立方体中必须有一个吸引盆 $\mathcal{N}(\mathbf{x})$，使得对 $\mathcal{N}(\mathbf{x})$ 中的所有初始状态向量 $\mathbf{x}(0)$，BSB 模型都收敛于 $\mathbf{x}$。为了使单位超立方体的每一个角点是一个可能的点吸引子，权值矩阵必须满足第四个条件(Greenberg，1988)：

- 权矩阵 $\mathbf{W}$ 是强对角优势的，表示为

$$w_{jj}\geq\sum_{i\neq j}|w_{ij}|+\alpha \qquad 对于\ j=1,2,\cdots,N \tag{14.75}$$

其中 α 是一个正的常数。

这里讨论的重点是：如果 BSB 模型的权值矩阵 $\mathbf{W}$ 只是对称的和正半定的，单位立方体中只有一些(不是所有)角点是点吸引子。为了使单位立方体中的所有角点是潜在的点吸引子，权矩阵 $\mathbf{W}$ 也必须满足式(14.75)，(14.75)当然蕴含式(14.74)。

聚类

BSB 模型的一个自然应用是聚类。这是因为单位超立方体的稳定角点作为有吸引盆的点吸引子，会把状态空间划分为相应的明确定义的区域。因此，BSB 模型可以用作一种无监督的聚类算法，其中单位超立方体的每一个稳定角点代表相关数据的一个“聚类”。由正反馈所提供的自放大(符合在第 8 章描述的自组织规则 1)是聚类性质的一个重要成分。 707

Anderson et al.(1990 b)描述用 BSB 模型聚类从而识别从不同发射器发射的雷达信号。在这个应用中，作为 BSB 模型运行基础的权值矩阵 $\mathbf{W}$ 用第 2 章描述的带误差修正学习的线性

联想器(联想记忆)进行学习。特别地，假设信息用一组 K 个训练向量表示，这些向量同它们自己的联系如下：

$$\mathbf{x}_k \to \mathbf{x}_k, \qquad k = 1,2,\cdots,K \tag{14.76}$$

以随机方式选定训练向量 $\mathbf{x}_k$。权值矩阵 $\mathbf{W}$ 按照误差修正算法(参看习题 13.9)

$$\Delta\mathbf{W} = \eta(\mathbf{x}_k - \mathbf{W}\mathbf{x}_k)\mathbf{x}_k \tag{14.77}$$

增加，其中 η 是学习率参数。学习刺激集 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_K$ 的目的是使线性联想器工作如下：

$$\mathbf{W}\mathbf{x}_k = \mathbf{x}_k, k = 1,2,\cdots,K \tag{14.78}$$

式(14.77)描述的误差修正算法在最小均方误差的意义下接近式(14.78)的理想条件。这个学习过程的最终效果是使线性联想器产生一组特征向量(由训练向量定义)，其特征值等于 1。

为了实施雷达(信号)聚类，BSB 模型用带误差修正学习的线性联想器来构造权矩阵 $\mathbf{W}$，并完成下面的计算(Anderson et al.， 1990)：

$$\mathbf{x}(n+1) = \varphi(\gamma\mathbf{x}(n) + \beta\mathbf{W}\mathbf{x}(n) + \delta\mathbf{x}(0)) \tag{14.79}$$

此式和式(14.63)和式(14.64)所描述的 BSB 算法有细微的差别。差别在两方面：

- 在第一项 $\gamma\mathbf{x}(n)$ 中的衰减常数 γ 使当前状态轻微衰减。假定 γ 是一个比 1 小的正常数，误差最终衰减到 0。
- 第三项 $\delta\mathbf{x}(0)$ 是为了保持初始状态向量 $\mathbf{x}(0)$ 一直出现；它有限制 BSB 模型的可能状态的作用。

708

BSB 模型的重复迭代导致由具有最大特征值的权值矩阵 $\mathbf{W}$ 的特征向量所支配的行动。因此，线性联想器学会了向量 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_K$。BSB 模型的聚类能力来源于：信号相关的特征向量与大的特征值相对应，在模型中由正反馈进行放大，因此在大量迭代之后便支配模型的状态。另一方面，噪声相关的特征向量经常与小的特征值相对应。因此，对 BSB 模型的状态有一个逐渐减小的影响，只要接受的信噪比足够高。

在一个雷达监视环境中，环境中发射器运行的细节描述是未知的。在几分之一秒内接受成千上万的雷达脉冲进行处理。因此不缺数据，难点是怎样使数据有意义。BSB 模型利用其内在的聚类属性通过学习雷达环境的微波结构来提供帮助。聚类形成在 BSB 模型的点吸引子周围(即单位超立方体的稳定角点)，每个点吸引子代表一个特定的发射器。这样，BSB 模型就可以识别一个特定发射器所产生的脉冲。

14.11 计算机实验Ⅱ

对于一个包含两个神经元的 BSB 模型，图 14-23 给出试验的结果。2×2 权值矩阵 $\mathbf{W}$ 定义为

$$\mathrm{W} = \begin{bmatrix} 0.035 & -0.005 \\ -0.005 & 0.035 \end{bmatrix}$$

此权矩阵是对称正定的，并满足式(14.75)。

图 14-23 的四个不同部分分别对应初始状态 $\mathbf{x}(0)$ 的四种不同的赋值，如下所示：

(a)$\mathbf{x}(0) = [0.1 \quad 0.2]^T$

(b)$\mathbf{x}(0) = [-0.2 \quad 0.3]^T$

(c)$\mathbf{x}(0) = [-0.8 \quad -0.4]^T$

(d)$\mathbf{x}(0) = [0.6 \quad 0.1]^T$

图中阴影区域是标志模型的四个吸引盆。该图清晰地阐明当模型的初始状态在一个特定的吸引盆时，模型固有动力学驱使权值矩阵 $\mathbf{W}(n)$随着迭代次数 n 的增加而增加，直到网络状态 $\mathbf{x}(n)$终止在一个固定点吸引子(即一个 2×2 正方形的角点)，此吸引子属于那个吸引盆。特别有趣的是图 14-23d 中的轨迹：初始条件 $\mathbf{x}(0)$在第一象限，然而轨迹在第四象限终止于角点$(+1,-1)$，因为那就是合适的吸引盆中点吸引子所在的地方。

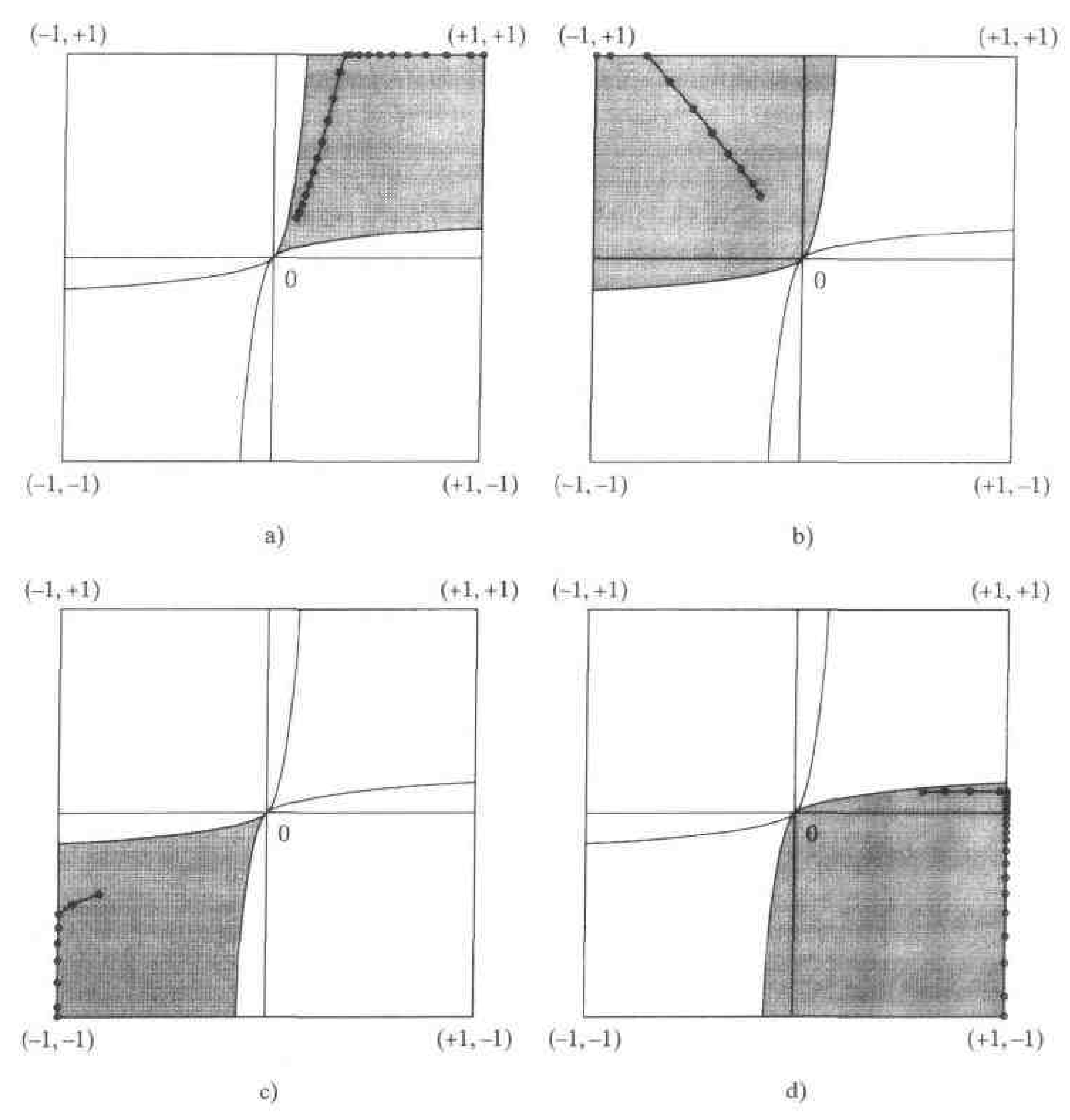


图 14-23 BSB 模型计算机实验的轨线；从 a)到 d)的结果对应于不同初始条件

14.12 奇异吸引子和混沌

到目前为止，在我们讨论的神经动力学中，集中于由固定点吸引子所刻画的非线性动力学系统的行为。在这一节考虑一种称为奇异吸引子的另一类吸引子，它们刻画阶数高于 2 的某种非线性动力学系统。 709

一个奇异吸引子表现出高度复杂的混乱行为。使研究奇异吸引子和混沌特别有趣的是：因为系统运行是由固定规则所支配的，所以系统是*确定的*。然而这样一个只有少数几个自由度的系统却有如此复杂的行为以至于它看起来是随机的。确实，随机性在以下意义上是基本

的：一个混沌(chaos)时间序列的二阶统计性似乎显示它是随机的。然而，不像一个真正的随机现象，一个混沌系统所展示的随机性并不随着收集信息的增加而减少。原则上，一个混沌系统未来的行为完全由它的过去所决定。但实际上，初始条件选择的任何不确定性，无论是多么小，随着时间将指数增加。这样即使一个混沌系统的动态行为在短期内可以预测，却不可能预测系统的长期行为。因此，一个混沌时间序列表现这样一种矛盾：它的产生是由一个确定动态系统支配的，然而它看起来却是随机的。一个混沌现象的这种属性最初是由Lorenz在发现一种吸引子时所强调的，并以他的名字命名(Lorenz, 1963)。

710

在一个非线性动态系统中，当吸引子中具有相近初始条件的不同轨迹随着时间增加而逐渐分离时，我们就说系统具有一个*奇异吸引子*(strange attractor)，并且说系统本身是*混沌的*(chaotic)。换句话说，使得一个吸引子奇异的本质属性是*对初始条件的敏感依赖*。这里，敏感性意味着如果两个相同的非线性系统开始于稍有差别的初始条件，即分别为 $\mathbf{x}$ 和 $\mathbf{x}+\boldsymbol{\varepsilon}$，这里 $\boldsymbol{\varepsilon}$ 是一个非常小的量，它们的动态状态在状态空间中会相互散开，并且它们的间隔平均而言将按指数增加。

混沌动力学的不变特征

两个主要特征分数维数(fractal dimensions)和Lyapunov指数，已经成为一种混沌过程的分类器。分数维刻画一个奇异吸引子的几何结构。术语"分数"(fractal)是由Mandelbrot(1982)提出的。不像整数维数(如二维平面、三维空间)，分数维数并不是整数。对于Lyapunov指数，它们描述吸引子的轨道如何随动态系统的演化而运动。这两个混沌动态系统的不变特征将在下面讨论。术语"不变"表明：一个混沌过程的分数维数和Lyapunov指数在该过程坐标系统的光滑非线性变换下保持不变(Abarbanal 1996)。

分数维数

考虑一个奇异吸引子，它是 d 维状态空间的动力学由

$$\mathbf{x}(n+1)=\mathbf{F}(\mathbf{x}(n)),\ n=0,1,2,\cdots \tag{14.80}$$

描述，它是式(14.2)的离散时间形式。通过设置 $t=n\Delta t$，这很容易看出，其中 Δt 是采样周期。假定 Δt 足够小，我们可以相应地设置

$$\frac{d}{dt}\mathbf{x}(t)=\frac{1}{\Delta t}[\mathbf{x}(n\Delta t+\Delta t)-\mathbf{x}(n\Delta t)]$$

711

这样，我们可以得到式(14.2)的离散时间形式如下：

$$\frac{1}{\Delta t}[\mathbf{x}(n\Delta t+\Delta t)-\mathbf{x}(n\Delta t)]=\mathbf{F}(\mathbf{x}(n\Delta t))\quad \text{对很小的 }\Delta t$$

为了表示方便，令 $\Delta t=1$ 并对项进行重新排列，得到

$$\mathbf{x}(n+1)=\mathbf{x}(n)+\mathbf{F}(\mathbf{x}(n))$$

它能写成式(14.80)的形式，只要简单地重新定义向量值函数 $\mathbf{F}(\cdot)$。

回到式(14.80)，假定我们在吸引子的轨道上或附近的一个位置 $\mathbf{y}$ 处构造半径为 r 的小球。那么，我们对吸引子可以定义点的*自然分布*(natural distribution)如下：

$$\rho(\mathbf{y})=\lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{y}-\mathbf{x}(n)) \tag{14.81}$$

其中 $\delta(\cdot)$ 是 d 维delta函数，N 是数据点的个数。注意 N 在用法上的变化。自然分布 $\rho(\mathbf{y})$ 对

一个奇异吸引子扮演的角色就像一个概率密度函数对一个随机变量那样。相应地，我们可以随动态系统演化定义函数 $f(\mathbf{y})$的不变量 $\bar{f}$ 为多重积分

$$\bar{f} = \int_{-\infty}^{\infty} f(\mathbf{y})\rho(\mathbf{y})d\mathbf{y} \tag{14.82}$$

一个感兴趣的函数 $f(\mathbf{y})$是使我们能衡量当小球半径 r 趋向于 0 时小球内的点的数目如何变化。注意 d 维球所占的空间体积正比于r^d，因此，通过观察在状态空间中吸引子上的点的密度在小距离范围内如何变化，我们可以了解吸引子的维数。

球的中心 $\mathbf{y}$ 和在时刻 n 时的点 $\mathbf{x}(n)$之间的 Euclid 距离是 $\|\mathbf{y}-\mathbf{x}(n)\|$。因此，如果 $\|\mathbf{y}-\mathbf{x}(n)\|<r$，或等价地 $r-\|\mathbf{y}-\mathbf{x}(n)\|>0$，则点 $\mathbf{x}(n)$在半径为 r 的球内。因此，在所描述情况下的函数 $f(\mathbf{x})$可以写成一般形式

$$f(\mathbf{x}) = \left(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{y}-\mathbf{x}(k)\|)\right)^{q-1} \tag{14.83}$$

712

其中 q 是一个整数，$\theta(\cdot)$是由

$$\theta(z) = \begin{cases}1, z>0\\0, z<0\end{cases}$$

定义的 Heaviside 函数。

将式(14.81)和(14.83)代入(14.82)，得到一个新的依赖于 q 和 r 的函数，所示：

$$C(q,r) = \int_{-\infty}^{\infty}\left(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{y}-\mathbf{x}(k)\|)\right)^{q-1}\left(\frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{y}-\mathbf{x}(n))\right)d\mathbf{y}$$

因此，利用 delta 函数的筛选(sifting)性质，也就是对某些函数 $g(\cdot)$的关系

$$\int_{-\infty}^{\infty} g(\mathbf{y})\delta(\mathbf{y}-\mathbf{x}(n))d\mathbf{y} = g(\mathbf{x}(n))$$

并交换求和顺序，可以重新定义函数 $C(q,r)$如下：

$$C(q,r) = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{x}(n)-\mathbf{x}(k)\|)\right)^{q-1} \tag{14.84}$$

函数 $C(q,r)$被称为相关函数(correlation function)[7]，它用来度量吸引子上两点 $\mathbf{x}(n)$和 $\mathbf{x}(k)$以距离 r 隔开的概率。在式(14.84)的定义中数据点的总数 N 假定很大。

相关函数 $C(q,r)$是吸引子本身的不变量。虽然如此，在实际中我们集中在 r 很小时 $C(q,r)$的行为。这个极限行为由

$$C(q,r) \simeq r^{(q-1)D_q} \tag{14.85}$$

描述，其中 D_q 称为吸引子的分数维数，假定它是存在的。在式(14.85)两边取对数，得到 D_q 的正式定义

$$D_q = \lim_{r\to 0}\frac{\log C(q,r)}{(q-1)\log r} \tag{14.86}$$

然而，由于通常仅有有限个数据点，半径 r 必须恰好足够小，使得有足够的点落在球内。对一个给定的 q，可以根据 $C(q,r)$作为 $\log r$ 的线性函数的斜率确定分数维数 D_q。

对 $q=2$，分数维数 D_q 的定义具有一个适宜于可靠计算的简单形式。所得维数 D_2 被称为吸引子的相关维数(correlation dimension)(Grassberger and Procaccia, 1983)。相关维数反映固有动态系统的复杂性，并且限定描述该系统所需的自由度。

Lyapunov 指数

Lyapunov 指数是描述吸引子未来状态不确定性的统计量。更具体地，它们量化在移向吸引子时邻近轨道相互分离的指数速度。假定 $\mathbf{x}(0)$是初始条件，$\{\mathbf{x}(n), n=0,1,2\cdots\}$是相应的轨道。考虑从初始条件 $\mathbf{x}(0)$向和轨道相切的向量 $\mathbf{y}(0)$方向上的一个无穷小偏移，该向量的演化确定被扰动轨道$\{\mathbf{y}(n), n=0,1,2,\cdots\}$从未受扰动轨道$\{\mathbf{x}(n), n=0,1,2\cdots\}$的无穷小偏移的演化。特别地，比值 $\mathbf{y}(n)/\|\mathbf{y}(n)\|$ 定义轨道从 $\mathbf{x}(n)$的无穷小偏移。当 $\|\mathbf{y}(n)\| > \|\mathbf{y}(0)\|$ 时，比值 $\mathbf{y}(n)/\|\mathbf{y}(0)\|$ 为无穷小偏移的增长因子；当 $\|\mathbf{y}(n)\| < \|\mathbf{y}(0)\|$ 时，它为无穷小偏移的缩减因子。对初始条件 $\mathbf{x}(0)$和初始偏移 $\alpha_0 = \mathbf{y}(0)/\|\mathbf{y}(0)\|$，Lyapunov 指数被定义为：

713

$$\lambda(\mathbf{x}(0),\alpha) = \lim_{n\to\infty}\frac{1}{n}\log\left(\frac{\|\mathbf{y}(n)\|}{\|\mathbf{y}(0)\|}\right) \tag{14.87}$$

一个 d 维混沌过程共有 d 个 Lyapunov 指数，可为正、负或 0。正的 Lyapunov 指数说明状态空间中一轨道的不稳定性。换句话说，正的 Lyapunov 指数导致混沌过程对初始条件的敏感性。另一方面，负的 Lyapunov 指数控制轨道中瞬态的衰减。一个为 0 的 Lyapunov 指数表明用以产生混沌的固有的动态系统可用一个联立的非线性微分方程组描述，即是说该混沌过程是一个流。在 d 维状态空间中体积依 $\exp(L(\lambda_1+\lambda_2+\cdots+\lambda_d))$变化，这里 L 是未来的时间步数。因此对一个耗散过程，所有 Lyapunov 指数之和必须是负数。这是状态空间的体积要随时间增加而缩减所必须满足的条件，它是物理实现的一个要求。

Lyapunov 维数

给定 Lyapunov 谱$\lambda_1,\lambda_2,\cdots,\lambda_d$，Kaplan and Yorke(1979)提出了一个奇异吸引子的 Lyapunov 维数定义如下：

$$D_L = K + \frac{\sum_{i=1}^{K}\lambda_i}{|\lambda_{K+1}|} \tag{14.88}$$

其中 K 是满足下列两个条件的整数：

$$\sum_{i=1}^{K}\lambda_i > 0 \text{ 和 } \sum_{i=1}^{K+1}\lambda_i < 0$$

通常，Lyapunov 维数 D_L 和相关维数 D_2 的大小大体相同。这是混沌过程的一个重要属性。也就是说，虽然 Lyapunov 维数和相关维数是用完全不同的方式定义，但对一个奇异吸引子，它们的值是非常接近的。

混沌过程的定义

在整个这一节中我们说到了混沌过程，但没有正式定义它。根据我们对 Lyapunov 指数的了解，可以给出如下定义：

一个混沌过程是由一个非线性确定系统产生的，它至少有一个正的 Lyapunov 指数。

至少有一个正的 Lyapunov 指数是“对初始条件敏感性”成立的必要条件，对初始条件敏感是一个奇异吸引子的特点。

最大的 Lyapunov 指数也定义一个混沌过程的可预测范围。特别地，一个混沌过程的短期可预测性近似等于最大 Lyapunov 指数的倒数(Abarbanal,1996)。

14.13　动态重构

714

动态重构可以定义为映射的辨识，该映射对一个未知的 m 维动态系统提供模型。这里，我们的兴趣是对一个已知为混沌的物理系统产生的时间序列进行动态建模。换句话说，给定一时间序列 $\{y(n)\}_{n=1}^{N}$，我们希望建造一个模型来捕获产生可观察 $y(n)$ 的潜在动力学。如我们在前面一节开头指出的那样，N 代表样本大小。动态重构的主要动机是从这样一个时间序列中得到实际意义，从而绕开对潜在动力学的详细数学知识的需要。感兴趣的系统一般太复杂以至于不能用数学方式刻画它。我们仅有的可用信息包含在对系统的一个可观测量进行测量所得到的时间序列内。

动态重构理论[8]最基本的结果是一个称为*延迟-嵌入*(delay-embedding)定理的几何定理，该定理是由 Tokens(1981)提出的。Takens 考虑一个无噪声系统，集中于*延迟坐标映射*(delay coordinate map)或*预测*(predictive)*模型*，映射或模型是由表示动态系统的一个可观测量所表示的时间序列构造的。特别地，Takens 证明：如果动态系统和可观测量是一般的(generic)，那么从一个 d 维光滑紧流形到 $\mathbb{R}^{2d+1}$ 的延迟坐标映射在该流形上是微分同胚(diffeomorphism)，这里 d 是动态系统状态空间的维数(微分同胚在 15.3 节讨论)。

为了用信号处理术语对 Takens 定理作解释，首先考虑一个未知的动态系统，该系统在离散时间的演化由非线性差分方程

$$\mathbf{x}(n+1) = \mathbf{F}(\mathbf{x}(n)) \tag{14.89}$$

描述，其中 $\mathbf{x}(n)$ 是系统在时刻 n 的 d 维状态向量，$\mathbf{F}(\cdot)$ 是一个向量值函数。这里假定采样周期为 1。系统输出的时间序列 $\{y(n)\}$ 用状态向量 $\mathbf{x}(n)$ 定义如下：

$$y(n) = g(\mathbf{x}(n)) + \nu(n) \tag{14.90}$$

其中 $g(\cdot)$ 是标量值函数，$\nu(n)$ 表示加性噪声。噪声 $\nu(n)$ 解释为在观测 $y(n)$ 中的不完全和不精确的综合效果。式(14.89)和(14.90)描述动态系统的状态空间行为。根据 Takens 定理，多变量动态系统的几何结构当 $\nu(n)=0$ 时可以从新向量

715

$$\mathbf{y}_R(n) = [y(n), y(n-\tau), \cdots, y(n-(D-1)\tau)]^T \tag{14.91}$$

构成的 D 维空间中观察的 $y(n)$ 展现，其中 τ 是一个称为*归一化嵌入延迟*的正整数。也就是说，对不同的离散时间 n，给定观察值 $y(n)$，它和未知动态系统的一个可观察值(分量)有关，假定 $D \geqslant 2d+1$，使用 D 维向量 $\mathbf{y}_R(n)$ 动态重构是可能的，其中 d 是系统状态空间的维数。以后我们就称这个陈述为*嵌入-延迟定理*。对动态重构来说，条件 $D \geqslant 2d+1$ 是充分的但不是必要的。寻找合适 D 的过程称为*嵌入*。能够实现动态重构的最小的整数 D 称为*嵌入维数*，用 D_E 表示。

嵌入-延迟定理具有很强的意义：重建空间中点 $\mathbf{y}_R(n) \rightarrow \mathbf{y}_R(n+1)$ 的演化服从原始状态空间中未知动态系统 $\mathbf{x}(n) \rightarrow \mathbf{x}(n+1)$ 的演化。也就是说，不能观察的状态向量 $\mathbf{x}(n)$ 的许多重要属性可以在由 $\mathbf{y}_R(n)$ 定义的重建空间中毫无疑义地得到。然而，为了获得这个重要结果，我们需要嵌入维数 D_E 和归一化嵌入延迟 τ 的*可靠估计*，如下综述：

- 充分条件 $D \geqslant 2d+1$ 使得解除吸引子一个轨道的自相交成为可能，这是出现在轨道

投影到低维数时出现的问题。嵌入维数 D_E 可以小于 $2d+1$。推荐的过程就是从可观测数据直接估计 D_E。估计 D_E 的可靠方法在 Abarbanal(1996)中描述的假最近邻方法。在此方法中，系统地考察数据点和它们的近邻，先在维数 $d=1$，然后 $d=2$，…，如此等等。我们借以确立明显近邻停止时的条件，是当添加更多元素到重构向量 $\mathbf{y}_R(n)$时“不被投影”，这样就获得对嵌入维数 D_E 的估计。

- 很不幸，延迟－嵌入定理并未提及归一化嵌入延迟 τ 的选择问题。事实上，只要可用时间序列无限长，它允许用任何的 τ。然而，实际上我们只能在有限长度 N 的观察数据上工作。选择 τ 的正确方法是认识到归一化嵌入延迟 τ 对 $y(n)$和 $y(n-\tau)$应足够大，使它们基本上独立，这样才能作为重建空间的坐标；但也不能使它们完全独立，以致没有任何联系。满足这个要求的最好办法就是选择特定的 τ 使得 $y(n)$和 $y(n-\tau)$之间的互信息获得它们第一个最小值(Fraser,1989)。互信息在第 10 章讨论。

递归预测

从前面讨论中知道，动态重构问题可以解释为恰当地表示信号动力学(嵌入步骤)和建造一个预测映射(识别步骤)。因此，实际上我们用下面的网络拓扑结构来进行动态建模。

- 短期记忆(例如延迟线记忆)结构实现嵌入，由此根据可观察的 $y(n)$和它的延迟形式来定义重建向量 $\mathbf{y}_R(n)$；参见式(14.91)。
- 训练作为单步预测器(如神经网络)的多输入单输出(MISO)自适应非线性系统，用它识别未知映射 f: $\mathbb{R}^D \to \mathbb{R}^1$，定义如下：

716

$$\hat{y}(n+1)=f(\mathbf{y}_R(n)) \tag{14.92}$$

式(14.92)描述的预测映射是动态建模的中心问题：一旦它被确定，演化 $\mathbf{y}_R(n)\to\mathbf{y}_R(n+1)$变成已知，由此确定未知演化 $\mathbf{x}(n)\to\mathbf{x}(n+1)$。

现在，我们没有一个严格的理论来帮助我们决定非线性预测器是否已成功地识别这个未知映射 f。在线性预测中，最小化预测误差的均方值可以得到一个精确的模型。然而，一个混沌时间序列不同。同一个吸引子的两个轨道在每次采样基础上都有很大的不同，所以最小化预测误差的均方值对一个成功的映射仅是必要条件而不是充分条件。

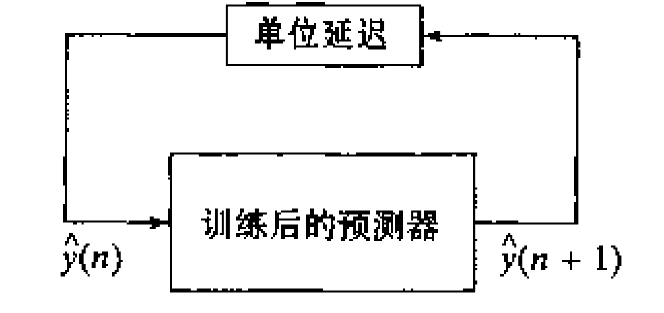


图 14-24 在混沌过程动态重构中用于迭代预测的单步预测器

动态不变量，即相关维数和 Lyapunov 指数，度量吸引子的全局属性，所以它们应该可以判断动态建模的成功与否。因此，*检验动态建模的一个实际方法是在奇异吸引子上挑选一点，然后反馈输出到其输入成为一个自治系统*，如 14-24 图所示。这样一个操作称为迭代预测或递归预测。一旦初始化完成，该自治系统的输出就是动态重构过程的一个实现。这当然要假定预测器开始时已被正确地设计。

我们说由图 14-24 描述的自治系统进行的动态重构是成功的，如果下面的两个条件成立(Haykin and Principe,1998)：

- *短期行为*。一旦初始化完成，在一段时间内图 14-24 中重建时间序列$\{\hat{y}(n)\}$紧紧跟随原来的时间序列$\{y(n)\}$，这段时间平均等于从过程的 Lyapunov 谱确定的可预测范围。

- 长期行为。从重建时间序列$\{\hat{y}(n)\}$计算的动态不变量和从原来的时间序列$\{y(n)\}$计算的动态不变量紧密地匹配。

为了判断重建动态系统的长期行为，需要估计(1)作为衡量吸引子复杂度的相关维数，(2)用于评价对初始条件的敏感性和估计 Lyapunov 维数的 Lyapunov 谱构成的框架；参看式(14.88)。Lyapunov 维数应该和相关维数的值相近。

递归预测的两种可能的形式

式(14.91)定义的重建向量 $\mathbf{y}_R(n)$的维数为 D_E，假定维数 D 和嵌入向量 D_E 相等。要实施嵌入的延迟线记忆的大小是 τD_E。但延迟线记忆仅要求提供 D_E 个输出(重建空间的维数)；也就是说，用 τ 个相等间隔的抽头表示稀疏连接。

另外，也可以把重建向量 $\mathbf{y}_R(n)$定义为一个完全的 m 维向量

$$\mathbf{y}_R(n) = [y(n), y(n-1), \cdots, y(n-m+1)]^T \tag{14.93}$$

其中 m 是一个整数，定义为

$$m \geqslant D_E \tau \tag{14.94}$$

717

第二种重建向量 $\mathbf{y}_R(n)$的形式比式(14.91)提供的形式对可预测模型提供更多的信息，因此可能产生一个更精确的动态重构。然而，这两种形式有一个共同的特点：它们的组成都由嵌入维数 D_E 的知识惟一定义。在任何情况下，明智的方法是用最小允许的值 D，也就是 D_E，来最小化加性噪声 $\nu(n)$对动态重构质量的影响。

动态重构是一个不适定的过滤问题

由于以下一个或多个原因，动态重构实际上是一个不适定的逆问题(逆问题适定的条件在第 5 章中讨论)。首先，由于一些未知的原因存在条件可能被破坏。第二，在可观察时间序列上的信息不足以惟一重建非线性动态系统；因此，惟一性标准被破坏。第三，不可避免地出现加性噪声和观察时间序列的某种不精确都会增加动态重构的不确定性。特别地，如果噪声水平太高，连续性标准也可能被破坏。那么怎么使动态重构问题适定呢？答案在于把包含关于输入－输出映射的先验知识的某种形式作为主要要求。换句话说，在预测模型的设计中，为了解决动态重构问题需要引入某种形式的限制(例如输入－输出映射的光滑性)。满足这个要求的有效方法是用 Tikhonov 的*正则化理论*，这也在第 5 章讨论。

另一个需要考虑的问题是预测模型以足够精度解决逆问题的能力。在这个背景下，用神经网络建造预测模型是合适的。特别地，多层感知器或径向基函数网络的通用逼近特性意味着我们利用具有适当规模的这种或那种神经网络可以注意重建精度的问题。另外，由于刚才说明的理由我们需要正则化的解决方法。理论上，多层感知器和径向基函数网络都适宜正则化的使用；实际上，我们发现在径向基函数网络中包括正则化理论作为它们设计的整体部分，在数学上易于处理。所以，在下一节描述的计算机实验中，集中以正则化的径向基函数(RBF)网络(在第 5 章描述)解决动态重构问题。

14.14 计算机实验Ⅲ

为了阐明动态重构的思想，我们考虑有三个联立常微分方程组的系统。该系统由 Lorenz 718

(1963)从低压大气热对流的偏微分方程组的 Galerkin 近似抽象而来，它成为测试非线性动态系统思想的一个主要方程组。Lorenz 吸引子的方程组为

$$
\begin{aligned}
\frac{dx(t)}{dt} &= -\sigma x(t) + \sigma y(t) \\
\frac{dy(t)}{dt} &= -x(t)z(t) + rx(t) - y(t) \\
\frac{dz(t)}{dt} &= x(t)y(t) - bz(t)
\end{aligned}
\tag{14.95}
$$

其中 σ，r 和 b 是无量纲参数。这些参数的典型值是 $\sigma = 10$，$b = 8/3$，$r = 28$。

图 14-25 显示在两个具有 400 个中心的 RBF 网络上，使用基于 Lorenz 吸引子的 $x(t)$ 分量的带噪声时间序列实施迭代预测的结果。信噪比是 25 分贝。在图 14-25a 中，网络的设计被

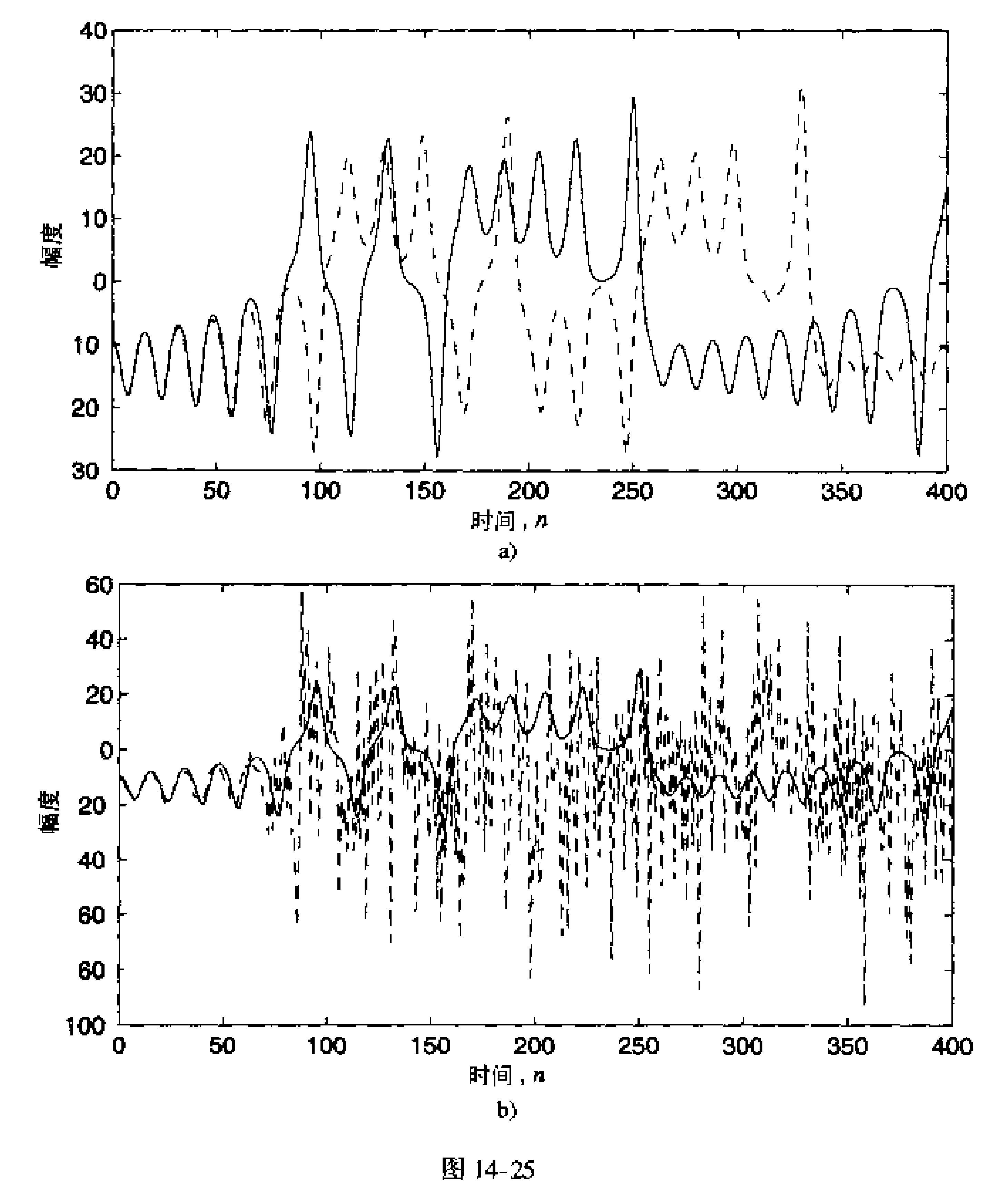


图 14-25

a)对 SNR = +25 分贝的 Lorenz 数据的正则化迭代预测($N = 400$，$m = 20$)

b)对 SNR = +25 分贝的 Lorenz 数据的无正则化迭代预测($N = 400$，$m = 20$)

在 a)和 b)中实线为实际的混沌信号，虚线为重构信号

正则化。在图 14-25b 中，网络设计未被正则化。图 14-25 的这两部分清楚地表明正则化的重要性。在没有正则化的情况下，图 14-25b 中显示的动态重构问题的解决方法是不能接受的，因为它不能近似 Lorenz 吸引子的真正轨迹；非正则化系统仅仅是一个预测器。另一方面，图 14-25a 中表示的动态重构问题的解决方法已经学会动态系统，因为根据迭代预测的网络输出和 Lorenz 吸引子在短期的真正轨迹非常接近。这一点为表 14-5 记录的结果证实，其中我们总结了三种情况下的 Lorenz 数据。

(a)无噪声 Lorenz 系统

(b)信噪比 SNR = 25 分贝的 Lorenz 系统

(c)用图 14-25a 的带噪声 Lorenz 时间序列的重建数据

用带噪声数据的重建数据的不变量和用无噪声 Lorenz 数据的重建数据不变量相近。偏差的绝对值是由于嵌入重建吸引子的噪声的残留影响以及估计程序的不精确。图 14-25 清楚地显示动态建模比预测有更多东西。这幅图以及很多不包括在这里的其他图像都显示出正则化 RBF 的解对迭代预测过程所用的吸引子上的初始化点的鲁棒性。

从图 14-25a 使用正则化得来的下面两点观察，是值得特别注意的：

1. 图 14-25a 的重建时间序列的短期可预测性是大约 60 个样本。从无噪声 Lorenz 吸引子的 Lyapunov 谱计算的理论可预测值是 100 个样本。试验和无噪声 Lorenz 吸引子的预测范围的偏差仅仅显示用来实施动态重构的实际数据里面存在噪声。从重建数据计算的理论可预测值范围是 61(表 14-5)，这非常接近短期可预测性的试验观察值。

719

2. 一旦超出短期可预测性的期限，用 14-25a 中的重建时间序列开始偏离真正 Lorenz 吸引子的无噪声实现。这基本上是混沌动力学的一个现象，也就是对初始条件的敏感性。像前面提到的那样，对初始条件的敏感性是混沌的一个标志。

表 14-5 用 Lorenz 系统的动态重构试验的参数小结

(a)无噪声 Lorenz 系统

使用样本数：35 000

1. 归一化嵌入延迟，$\tau = 4$

2. 嵌入维数，$D_E = 3$

3. Lyapunov 指数：

$\lambda_1 = 1.569\,7$

$\lambda_2 = -0.031\,4$

$\lambda_3 = -22.305\,4$

4. 可预测范围 $\simeq$ 100 个样本

720

(b)有噪声 Lorenz 系统：25 分贝 SNR

使用样本数：35 000

1. 归一化嵌入延迟，$\tau = 4$

2. 嵌入维数，$D_E = 5$

3. Lyapunov 指数：

$\lambda_1 = 13.268\,9$

$\lambda_2 = 5.856\,2$

$\lambda_3 = -3.144\,7$

$\lambda_4 = -18.008\,2$

$\lambda_5 = -47.057\,2$

（续）

4. 可预测范围$\simeq$12个样本
(c)用图14-25a的有噪声Lorenz数据重构的系统
产生样本数(递归地)：35 000
1. 归一化嵌入延迟，$\tau=4$
2. 嵌入维数，$D_E=3$
3. Lyapunov指数：
$\lambda_1=2.565\ 5$
$\lambda_2=-0.627\ 5$
$\lambda_3=-15.034\ 2$
4. 可预测范围$\simeq$61个样本

注意：所有的Lyapunov指数的单位为奈特/秒。如第10章讨论的那样，一个奈特是测量信息的一个自然单位。同样，在情形b中，噪声的影响是增加Lyapunov谱的大小和正Lyapunov指数的数量和大小。

m和λ的选择

输入层的大小m由式(14.94)决定。如以前解释的那样，推荐的方法是根据等号用最小的允许值m使得噪声对动态重构的影响最小化。

归一化嵌入延迟τ的估计值基本上不受噪声影响，适宜于较高的信噪比。相反，噪声对嵌入向量D_E的估计值有深刻的影响，这也符合直观。例如，对于无噪声Lorenz吸引子，相关维数是2.01。因此，我们可以选择嵌入维数$D_E=3$，这可由假近邻方法确认。归一化嵌入延迟为$\tau=4$。这样，用式(14.94)的等号可以得到动态重构的$m=12$。然而，对于一个有噪声的Lorenz吸引子，其中SNR = +25分贝，用假最近邻法得到$D_E=5$，用互信息法得到$\tau=$ 4。在式(14.94)中代入这些估计值并取等号，我们得到图14-25中有噪声动态重构的$m=$ 20。表14-5包含归一化嵌入延迟τ和嵌入维数D_E。

721

对于图14-25a中用到的正则化参数λ，它是用广义交叉确认(generalized cross-validation, GCV)方法由训练数据得到的，这种方法在第5章中讨论。图14-25a中所用的λ值，由GCV方法计算，根据数据的不同在最小值10^{-14}和最大值10^{-2}之间变化。

14.15 小结和讨论

这一章的很多材料都是在讨论Hopfield模型和BSB模型，它们都是作为植根于神经动力学的联想记忆的例子。这两个模型有下面一些共同特点：

- 它们都使用正反馈。
- 它们都有能量(Lyapunov)函数，固有的动力学以迭代方式使能量函数最小化。
- 它们都用Hebb学习规则进行自组织学习。
- 它们都能利用吸引子动力学进行计算。

很自然，它们各自的应用领域是不同的。

BSB模型固有的聚类能力使它很好地用在数据表示和概念形成上。BSB模型最有趣的应用可能是作为网络的网络(network of networks)的一个基本计算单元，网络的网络作为描述人脑内系统组织的不同层次的一个合理模型(Anderson and Sutton 1995)。在这个模型中，计算单

元构成分布在二维阵列中的局部网络，因此用术语“网络的网络”。不是从一列到另一列进行平均激活通信，这些局部网络设计为通过激活模式(向量)和其他局部网通信。在常规的神经网络中神经元之间有权值相连，与之类似的是现在我们用一组交互(interaction)矩阵来描述两个局部网络中吸引子之间的耦合。局部网络基于它们的内部连接形成聚类和层次使得它们的结构(anatomical)连接是稀疏的。也就是说，局部网络在内部的连接比它们之间的连接更稠密。然而，聚类之间的功能连接是富于动态的，这部分起因于局部网络之间的时间相关激活。

对比之下，Hopfield 模型可以用来解决下列计算问题：

722

1. *按内容寻址存储*，它涉及部分或失真的模式呈现给网络以检索存储的模式。在这个应用中，一般过程是利用基于 McCulloch-Pitts 神经元(即使用硬限制激活函数)的离散 Hopfield 模型。从计算的角度看，建造一个按内容寻址存储是很平凡的。然而一个按内容寻址存储的 Hopfield 网络是非常重要的，因为它以全新的方式阐明*动力学和计算之间的联系*。特别地，Hopfield 模型展示和神经生物学有关的下列属性：

- 模型的动力学在一个高维状态空间由大量吸引子支配。
- 一个感兴趣的点吸引子(即基础记忆)的位置，可以通过仅仅使用该吸引子位置的不精确描述初始化模型以及允许动态地演化模型状态到最近点吸引子来确定。
- 学习(即模型自由参数的计算)是按 Hebb 规则学习进行的。另外，这种学习机制允许新的点吸引子按希望的那样插入模型。

2. *组合最优化问题*，这类问题被数学家称为最难的一类。这类最优化问题包括经典的*旅行商问题*(traveling salesman problem, TSP)。给定一定数量城市的位置，假定在一个平面上，问题是找到最短的路径旅游完所有城市并返回出发点。TSP 问题陈述起来很简单，但却很难解决。除了计算每条可能路径的长度并选择最短路径外，没有其他已知的找最优路径的方法。TSP 问题是 *NP-完全的*(Hopcroft and Ullman, 1979)。在一篇开创性的文章中，Hopfield and Tank(1985)阐述基于联立一阶微分方程组的模拟网络怎样给出 TSP 问题的解。具体地，网络的权值由旅行中访问的城市间距离决定，该问题的最优解是神经动力学方程(14.20)的一个固定点。在此处遇到的困难就是将组合最优化问题映射到连续(模拟)Hopfield 网络上。网络使一个能量(Lyapunov)函数最小化，然而通常的组合优化问题要求满足一些硬的约束条件下使目标函数最小(Gee et al., 1993)。如果违反这些限制中的任何一个，则认为解是无效的。早期的映射过程是以特别方式建造的 Lyapunov 函数为基础的，通常用一项表示一个约束，由

$$E = E^{opt} + c_1 E_1^{cns} + c_2 E_2^{cns} + \cdots \tag{14.96}$$

723

表示。第一项 E^{opt} 是被最小化的目标函数(如 TSP 路径的长度)；它由当前的问题决定。剩余的项 $c_1 E_1^{cns}$，$c_2 E_2^{cns}$，…代表惩罚函数，它们的最小化满足*约束条件*。标量 c_1，c_2，…是赋予每个惩罚函数 E_1^{cns}，E_2^{cns}，…的常数权值。不幸的是，式(14.96)中 Lyapunov 函数的许多项都互相干扰，并且 Hopfield 网络的成功与否对 c_1，c_2，…的值非常敏感(Gee et al., 1993)。因此毫不奇怪，网络经常产生大量无效的解(Wilson and Pawley, 1988; Ansari and Hou, 1997)。在 Gee(1993)中列出用连续的 Hopfield 网络作为工具解决组合优化问题的一些基本问题，其中报告的主要发现可以概述如下：

- 给一个用二次0-1规划表示的组合优化问题，如像在旅行商问题中那样，网络有直接的方法来解决这个问题，求出的解不违反问题的任何约束条件。
- 基于复杂性理论和数学规划的结果，除了当问题的约束条件有可能产生整型多面体(integral polytope)的特殊属性外，证明不可能迫使网络收敛于一个有效的、可解释的解。用几何术语来说，一个多面体，即一个有界的多面体(bounded polyhedron)，我们说它是个*整型多面体*，如果它的所有顶点都是0-1点。即使处理整型多面体时，如果目标函数 E^{opt} 是二次的，则问题是NP-完全的，并不能保证网络能产生最优解。这类问题包含TSP问题。不过，如果给出对这个解的下降过程的性质，可以找到一个有效解，而且所得的解有很大的机会是值得信赖的。

本章考虑的Hopfield模型，在它的神经元之间使用对称连接。这样一个结构的动力学和梯度下降动力学类似，由此保证能收敛到一固定点。然而，人脑的动力学在两个重要方面和Hopfield模型不同：

- 人脑内的神经元连接是非对称的。
- 人脑中观察到振荡的和复杂的非周期性的行为。

实际上，正是因为人脑的这些特点，在Hopfield模型之前关于非对称网络[9]的研究兴趣已有很长历史了。

如果我们放弃对称性的限制，下一个最简单的模型是*兴奋-抑制网络*，它的神经元分为两个群体：一种是只有兴奋性输出，另一种只有抑制性输出。这两种类型神经元之间的连接是反对称的。然而，同种类型神经元之间的连接是对称的。在Seung et al.(1998)中考虑了这种网络的动力学。那里的分析利用兴奋-抑制网络和*梯度下降-梯度上升动力学*之间内在的相似性。这里运动方程在某些状态变量是梯度下降的，对另一些是梯度上升的。结果，不像梯度下降动力学刻画的Hopfield模型，Seung et al.(1998)所考虑模型的动力学能收敛到一个固定点或一个极限环中，这取决于网络参数的选择。因此，在Seung et al.(1998)中研究的非对称模型代表对对称的Hopfield模型的进一步发展。

724

注释和参考文献

[1] 一个*非自治*(nonautonomous)系统由状态方程

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{F}(\mathbf{x}(t), t), \mathbf{x}(t_0) = \mathbf{x}_0$$

定义。对一个非自治系统，向量域 $\mathbf{F}(\mathbf{x}(t), t)$ 依赖于时间 t。因此，不像自治系统那样，我们一般不置初始时间为0(Parker and Chua, 1989)。

[2] 一般地，除式(14.11)外一个非线性动态系统的全局稳定性还需要径向无界条件(Slotine and Li, 1991)

$$V(\mathbf{x}) \to \infty \qquad 当 \ \|\mathbf{x}\| \to \infty$$

成立。由具有sigmoid激活函数的神经网络构造的Lyapunov函数通常满足该条件。

[3] 我们给出一个吸引子的严格定义如下(Lanford 1981; Lichtenberg and Lieberman, 1992)：
状态空间的一个子集(流形) M 被称为一个吸引子，如果：

- M 关于流保持不变
- 在流中，M 周围有一个(开)邻域收缩到 M

- M 的所有部分都不是瞬态的
- M 不能被分成两个互不相交的不变片(piece)

[4] **集中点火(Integrate-and-Fire)神经元**

式(14.14)的加性模型并没完全抓住一个生物神经元的精髓。特别地，它忽略了动作电位里编码的时序信息；动作电位在第1章给出简要的定性描述。Hopfield(1994)描述一个动态模型，通过考虑一个集中点火神经元捕捉动作电位。这样一个神经元的运行由一阶微分方程

$$C\frac{d}{dt}u(t) = -\frac{1}{R}(u(t) - u_0) + i(t) \tag{1}$$

描述，其中 $u(t)$ = 神经元内部电位，C = 神经元周围细胞膜的电容，R = 细胞膜的漏阻(leakage resistance)，$i(t)$ = 由另一神经元注入当前神经元的电流，u_0 = 当 $i(t)$ 消失时神经元减少的电位。在每次内部电位 $u(t)$ 达到阈值时产生一个动作电位。

动作电位被看作是 Dirac delta(冲击)函数，表示为

$$g_k(t) = \sum_n \delta(t - t_{k,n}) \tag{2}$$

其中 $t_{k,n}$，$n = 1, 2, 3, \cdots$ 代表神经元 k 的激活动作电位的次数，这些次数由式(1)所定义。

流入神经元 k 的总电流的行为模型化为

$$\frac{d}{dt}i_k(t) = -\frac{1}{\tau}i_k(t) + \sum_j w_{kj}g_j(t) \tag{3}$$

725

其中 w_{kj} 为神经元 j 到神经元 k 的突触权值，τ 是神经元 k 的特征时间常数，函数 $g_j(t)$ 由式(2)定义。

式(14.4)的加性模型可看作是(3)的一个特例。具体地，忽略 $g_j(t)$ 尖峰(spiky)性质，而代之以 $g_j(t)$ 和一个光滑函数的卷积。这样做的理由如下，因为高度连接在一个合理的时间间隔内式(3)右边的总和会有许多项，并且我们只关心神经元 k 点火率的短期行为。

[5] Little 模型(Little, 1974; Little and Shaw, 1975)和 Hopfield 模型一样使用同样的权值。然而，它们不同之处在于 Hopfield 模型用异步(串行)动力学，而 Little 模型用同步(并行)动力学。相应地，它们展示不同的收敛性(Bruck, 1990; Goles and Martinez, 1990)。Hopfield 网络总是会收敛到一个稳定状态，而 Little 模型总是会收敛到一个稳定状态或长度至多为2的极限环。所谓"极限环"是指网络状态空间的长度小于或等于2的环。

[6] *非单调激活函数*

为了克服 Hopfield 模型作为按内容寻址存储的局限，文献中提出了各种各样的建议。也许到目前为止最有意义的改进是 Morita(1993)提出的，它应用于 Hopfield 模型的连续(模拟)形式。修改限制在一个神经元的激活函数 $\varphi(\cdot)$ 上，从而保持网络作为联想记忆的简单性。具体地，网络中的每个神经元的通常硬限制(hard-limiting)或 sigmoid 激活函数替换为非单调函数。在数学形式上，这个激活函数由两个因子的乘积定义，表示为

$$\varphi(v) = \left(\frac{1 - \exp(-av)}{1 + \exp(-av)}\right)\left(\frac{1 + \kappa\exp(b(|v| - c))}{1 + \exp(b(|v| - b))}\right) \tag{1}$$

其中 v 为诱导局部域。式(1)右边的第一项是连续 Hopfield 模型中常用的 sigmoid(双曲

正切)函数。第二项使激活函数$\varphi(v)$成为非单调的。第二项中的参数b和c是正的常数；参数κ通常是负的。在由Morita(1993)所做的试验中，各个参数赋值如下：

$$a = 50; b = 15, c = 0.5; \kappa = -1$$

根据Morita的研究，激活函数的形式和所用的参数并不苛刻；最本质的因素是激活函数的非单调属性。

Morita描述的一个按内容寻址存储器模型有两个有趣的性质(Yoshizawa et al., 1993)：

1. 对由N个神经元构成的网络，模型的存储容量约为0.3 N(对较大的N)，比常规Hopfield模型的相应值$N/(2\log N)$要大得多。

726

2. 模型没有出现任何伪状态(spurious state)。相反，当它不能恢复起一个正确的记忆模式时，网络状态被推进到一种混沌行为。混沌的概念在14.13节中讨论。

[7] 式(14.84)定义的相关函数$C(q,r)$的思想在统计上已知是从Rényi(1970)的工作得来的。然而用它去刻画一个奇异吸引子是在Grassberger and Procaccia(1983)中提出的。他们最初是讨论相关维数$q=2$时$C(q,r)$的应用。

[8] 从一个时间序列里用独立坐标来构建动态系统首先由Packard et al.(1980)提出。然而，这篇论文并没有给出证明，用的是"导数"嵌入而不是时间－延迟嵌入。时间－延迟嵌入或延迟坐标嵌入归功于Ruelle和Takens。特别地，1981年Takens发表了一篇在数学上很深刻的时间－延迟嵌入方面的文章，它应用于吸引子为曲面或类似环面；也可以参看Mañé(1981)在同一杂志上发表的同一主题的论文。Takens的论文对非数学家来说很难懂，Mañé的更难懂。延迟坐标映射的思想在Sauer et al.(1991)中得到提炼。在这篇论文中采用的方法是对Whitney(1936)和Takens(1981)的早期结果的综合和扩展。

[9] 将生物神经网络看成一个出现振荡行为和行波的非线性动态系统已有很长的历史(Wilson and Cowan 1972; Amari 1977a, 1977b; Amari and Arbib 1977)；也可以参看Carpenter et al.,(1987)的讨论。

习题

动力系统

14.1 对于状态向量$\mathbf{x}(0)$作为一个动态系统的平衡状态，重述Lyapunov定理。

14.2 验证图14-8a和14-8b的框图分别对应神经动力学方程(14.18)和(14.19)。

14.3 考虑一个一般的神经动力学系统，它依赖于未指定的内部状态参数、外部动态刺激和状态变量。系统由状态方程

$$\frac{dx_j}{dt} = \varphi_j(\mathbf{W}, \mathbf{u}, \mathbf{x}), \qquad j = 1, 2, \cdots, N$$

定义，其中矩阵$\mathbf{W}$代表系统的内部动态参数，向量$\mathbf{u}$代表外部动态刺激，$\mathbf{x}$是状态向量，它的第j个元素用x_j表示。对于$\mathbf{W}$，$\mathbf{u}$的值和在状态空间的某些运行区域$\mathbf{x}(0)$的值，假定系统的轨迹收敛到点吸引子(Pineda, 1988b)。讨论所描述的系统怎么能用于如下应用：

(a)连续映射器，$\mathbf{u}$是输入，$\mathbf{x}(\infty)$是输出

(b)自联想记忆，$\mathbf{x}(0)$是输入，$\mathbf{x}(\infty)$是输出

727

Hopfield模型

14.4 考虑5个神经元组成的Hopfield网络，它需要存储以下三个基本记忆：

$$\xi_1 = [+1, +1, +1, +1, +1]^T$$
$$\xi_2 = [+1, -1, -1, +1, -1]^T$$
$$\xi_3 = [-1, +1, -1, +1, +1]^T$$

(a)计算网络的 5×5 突触权值矩阵。

(b)用异步更新演示所有三个基本记忆 ξ_1，ξ_2，ξ_3 满足对齐条件。

(c)如 ξ_1 是有噪声的，它的第二个元素极性反转，研究网络的检索性能。

14.5 研究同步更新时习题 14.4 所描述 Hopfield 网络的检索能力。

14.6 (a)证明

$$\xi_1 = [-1, -1, -1, -1, -1]^T$$
$$\xi_2 = [-1, +1, +1, -1, +1]^T$$
$$\xi_3 = [+1, -1, +1, -1, -1]^T$$

也是习题 14.4 所描述的 Hopfield 网络的基本记忆。这些基本记忆和习题 14.4 中的基本记忆之间有什么关系?

(b)假定习题 14.4 中基本记忆 ξ_3 的第一个元素被损坏(即减少为 0)。确定 Hopfield 网络所产生的结果模式。比较这个结果和 ξ_3 的原始形式。

14.7 考虑由两个神经元构成的简单 Hopfield 网络，网络的突触权值矩阵为

$$\mathbf{W} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

每个神经元的偏置为 0，网络的四个可能状态是

$$\mathbf{x}_1 = [+1, +1]^T, \mathbf{x}_2 = [-1, +1]^T, \mathbf{x}_3 = [-1, -1]^T, \mathbf{x}_4 = [+1, -1]^T$$

(a)说明状态 $\mathbf{x}_2$ 和 $\mathbf{x}_4$ 是稳定的，而状态 $\mathbf{x}_1$ 和 $\mathbf{x}_3$ 成为极限环。用下面两个工具来说明：

1. 对齐(稳定性)条件
2. 能量函数

(b)刻画状态 $\mathbf{x}_1$ 和 $\mathbf{x}_3$ 的极限环的长度是多少?

14.8 在本题中，我们推导式(14.55)，它是计算按内容寻址存储器的 Hopfield 网络在几乎无错情况下的存储容量。 728

(a)误差函数的渐进行为可以近似描述为

$$\text{erf}(y) \simeq 1 - \frac{e^{-y^2}}{\sqrt{\pi} y}, \text{对大的 } y$$

用这个近似证明式(14.53)的条件概率可近似为

$$P(v_j > 0 \mid \xi_{\nu,j} = +1) \simeq 1 - \frac{e^{-\rho/2}}{\sqrt{\pi\rho}}$$

其中 ρ 是信噪比。证明稳定模式的概率相应近似为

$$p_{\text{stab}} \simeq 1 - \frac{N e^{-\rho/2}}{\sqrt{\pi\rho}}$$

(b)在(a)中公式 p_{stab} 的第二项是基本记忆中一个比特不稳定的概率。根据几乎没有错误的存储容量的定义，仅要求这一项较小是不够的；相反它和 $1/N$ 相比必须是小的，其中 N 是 Hopfield 网络的大小。证明信噪比必须满足条件

$$\rho > 2\log_e N + \frac{1}{2}\log_e(2\pi\rho)$$

(c)利用从(b)中得到的结果，证明为了大部分基本记忆能完全恢复所要求信噪比的最小允许值为

$$\rho_{min} = 2\log_e N$$

相应的 p_{stab} 是多少?

(d)用(c)的结果，证明

$$M_{max} \approx \frac{N}{2\log_e N}$$

正如式(14.55)中描述的那样。

(e)由(d)推导的存储容量公式是基于大部分基本记忆是稳定的。对无错误的存储容量给出一个更严格的定义，我们要求所有基本记忆都能被正确地检索。利用这后一个定义，证明能存储在Hopfield网络中的最大基本记忆的数目为(Amit 1989)

$$M_{max} \simeq \frac{N}{4\log_e N}$$

14.9　一个Hopfield网络的能量函数可表达为

$$E = -\frac{N}{2}\sum_{\nu=1}^{M} m_\nu^2$$

其中 m_ν 代表由

$$m_\nu = \frac{1}{N}\sum_{j=1}^{N} x_j \xi_{\nu,j}, \qquad \nu = 1,2,\cdots,M$$

定义的重叠，其中 x_j 是状态向量 $\mathbf{x}$ 的第 j 个元素，$\xi_{\nu,j}$ 是基本记忆 $\boldsymbol{\xi}_\nu$ 第 j 个元素，M 是基本记忆个数。

14.10　设计Hopfield网络用来存储两个基本记忆模式(+1, +1, -1, +1, +1)和(+1, -1, +1, -1, +1)。网络的突触权值矩阵如下：

729

$$\mathbf{W} = \begin{bmatrix} 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & -2 & 2 & 0 \\ 0 & -2 & 0 & -2 & 0 \\ 0 & 2 & -2 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a)矩阵 $\mathbf{W}$ 的特征值之和为0。为什么?

(b)网络的状态空间是 $\mathbb{R}^5$ 的一个子空间。详细说明这个子空间的结构。

(c)由基本记忆向量扩张的子空间记为 $\mathcal{M}$，矩阵 $\mathbf{W}$ 的零空间记为 $\mathcal{N}$。网络的固定点(稳定状态)和伪状态是什么?

(读者若希望了解这里描述的网络的动力学的一个更详细的描述，可以参考deSilva and Attikiouzzel(1992)的文章。)

14.11　图14-26显示一个非单调激活函数的分段线性形式。用这个近似形式的Hopfield网络的恢复动力学由

$$\frac{d}{dt}\mathbf{v}(t) = -\mathbf{v}(t) + \mathbf{W}\mathbf{x}(t), \mathbf{x}(t) = \text{sgn}(\mathbf{v}(t)) - k\mathbf{v}(t)$$

定义，其中 $\mathbf{v}(t)$ 是向量的诱导局部域，$\mathbf{W}$ 是突触权值矩阵，$\mathbf{x}(t)$ 是状态(输出)向量。$-k$ 是一个负的常数斜率。令 $\bar{\mathbf{v}}$ 是位于基本记忆 ξ_1 的象限内的网络平衡状态，并令

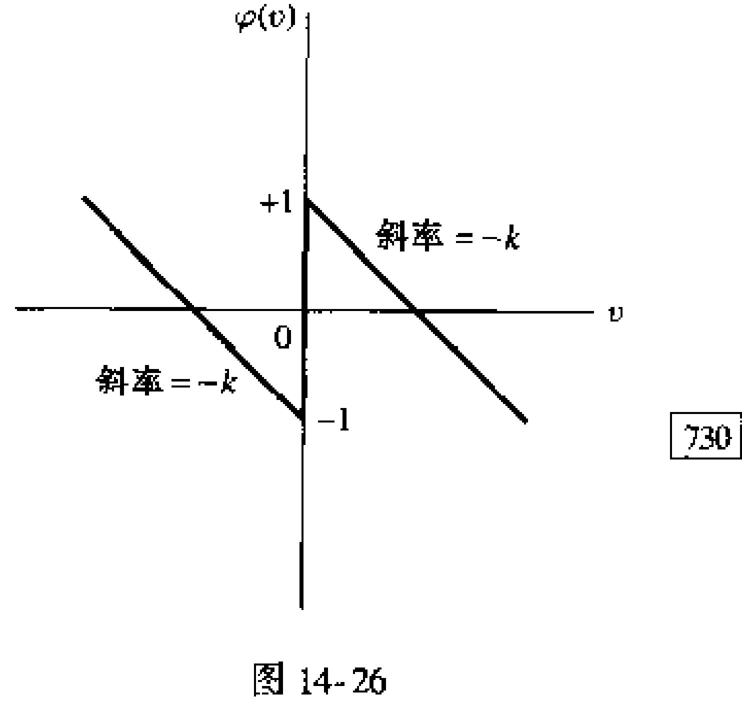


图 14-26

730

$$\mathbf{x} = \text{sgn}(\bar{\mathbf{v}}) - k\bar{\mathbf{v}}$$

证明 $\bar{\mathbf{x}}$ 由下面三个条件所刻画(Yoshizawa et al.，1993)：

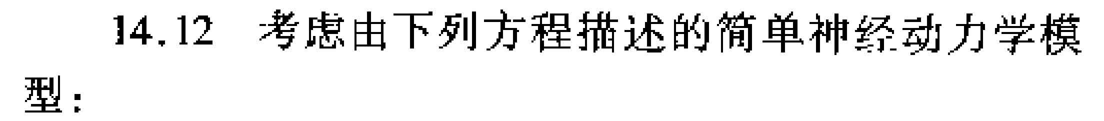
(a) $\sum_{i=1}^{N} \bar{x}_i \xi_{\mu,i} = 0, \qquad \mu = 2,3,\cdots,M$

(b) $\sum_{i=1}^{N} \bar{x}_i \xi_{1,i} = M$

(c) $\bar{x}_i < 1, \qquad i = 1, 2, \cdots, N$

其中 ξ_1，ξ_2，…，ξ_M 是存储在网络中的基本记忆，$\xi_{\mu,i}$ 是 ξ_μ 第 i 个元素，$\bar{x}_i$ 是 $\bar{\mathbf{x}}$ 的第 i 个元素，N 是神经元个数。

14.12 考虑由下列方程描述的简单神经动力学模型：



$$\frac{dv_j}{dt} = -v_j + \sum_i w_{ji}\varphi(v_i) + I_j, j = 1,2,\cdots,N$$

描述的系统总是会收敛到一个惟一的点吸引子，假定突触权值 w_{ji} 满足条件

$$\sum_j \sum_i w_{ji}^2 < \frac{1}{(\max|\varphi'|)^2}$$

其中 $\varphi' = d\varphi/dv_j$。考查这个条件的正确性。你可以参考论文(Atiya，1987)，该条件是从这篇文章导出的。

Cohen-Grossberg 定理

14.13 考虑式(14.57)定义的 Lyapunov 函数。如果式(14.59)至式(14.61)的条件满足，证明

$$\frac{dE}{dt} \leqslant 0$$

14.14 在 4.10 节，我们通过应用 Cohen-Grossberg 定理导出了 BSB 模型的 Lyapunov 函数。在推导式(14.73)时，省略了一些细节。请写出这些细节。

14.15 图 14-27 显示非单调激活函数的一个图形，该函数由 Morita(1993)提出，这在注释[6]中讨论过。这个函数在构造 Hopfield 网络时用于代替双曲线正切函数。Cohen-Grossberg 定理适用于这样构造的联想存储器吗？请说明你的理由。

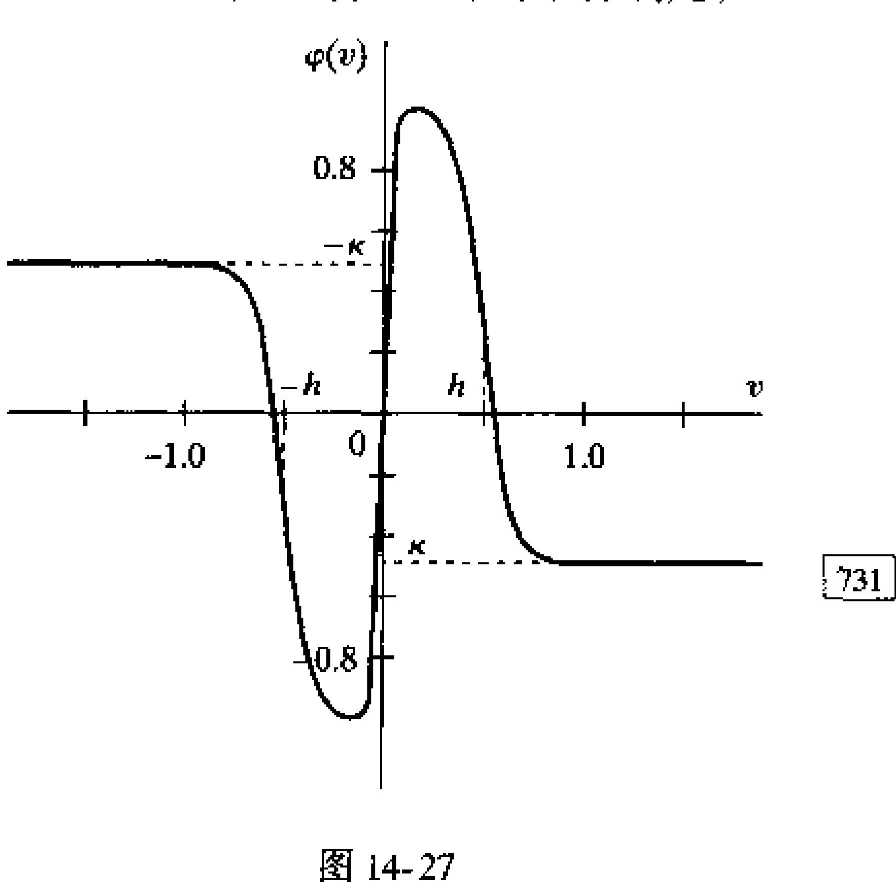


图 14-27

731

第15章 动态驱动的递归网络

15.1 简介

如在前一章提到的，*递归网络*是有一个或多个反馈环的神经网络。反馈可以是*局部的*或*全局的*。在这一章，我们继续研究有全局反馈环的递归网络。

给定多层感知器作为基本模块，应用全局反馈可以有不同的形式。反馈可以从多层感知器的输出神经元到输入层。还有另一种可能的全局反馈是从网络的隐藏神经元到输入层。当多层感知器有多个隐藏层时，全局反馈的可能形式甚至可以进一步扩大。要点是递归网络有丰富的结构布局。

基本上，递归网络有两个基本功能作用：

- *联想记忆*
- *输入－输出映射网络*

递归网络作为联想记忆在14章已经有详细叙述。这一章我们将研究作为输入－输出映射网络的用途。无论用途怎样，研究递归网络时特别关注的问题是它的*稳定性*；这个问题在第14章也考虑过。

由定义，映射网络的输入空间被映射到输出空间。对于这方面的应用，递归网络依时序响应外部应用的输入信号。因此我们在这一章里可以称递归网络为*动态驱动递归网络*。而且，反馈的应用使得递归网络获得*状态*表示，这使得它成为适应于不同应用的工具，例如非线性预测和建模，通信信道的自适应平衡，语音处理，设备控制以及汽车发动机的诊断。因此，递归网络提供第13章所说的动态驱动前馈网络的一种替代。 732

由于全局反馈的效益，它们实际可以运行得更好。使用全局反馈具有大大减少记忆需求的潜力。

本章的组织

本章分为四个部分：体系结构，理论，学习算法和应用。第一部分包含15.2节，讨论递归网络的体系结构。

第二部分包括15.3节至15.5节，处理递归网络的理论部分。15.3节描述状态空间模型以及相关的可控性和可观察性的问题。15.4节导出一个状态空间模型的等价模型，通称为有外部输入的非线性自回归的模型。15.5节讨论递归网络计算能力的一些理论问题。

第三部分包括15.6节至15.12节，讨论递归网络的学习算法和相关问题。开始在15.6节有一个对主题的综述。15.7节在第4章的材料基础上讨论通过时间的反向传播算法。15.8节讨论另一个流行算法：实时递归学习，15.9节对经典Kalman滤波理论进行简短综述，紧跟着15.10节描述解耦扩展的Kalman过滤算法。15.11节给出了后面这个算法用于递归学习的一个计算机实验。建立在梯度基础上的递归学习受到消失梯度问题的影响，15.12节对此有讨论。

第四部分也是本章最后一部分，包括15.13节和15.14节，讨论递归网络的两个重要应用。15.13节讨论系统辨识。15.14节讨论模型参考自适应控制。

在15.15节以一些最终评论结束章。

15.2 递归网络体系结构

如前面介绍所言，递归网络的结构布局有许多不同形式。本节讨论四种特殊结构，每一种着重于全局反馈的一种特殊形式[1]。它们有如下共同的特点：

- 它们都结合一个静态多层感知器或其中某些部分。
- 它们都利用多层感知器的非线性映射能力。

输入–输出递归网络

733 图15-1显示由一个多层感知器的自然推广而得到的通用递归网络模型。模型有一个输入被应用到有 q 个单元的抽头延迟线记忆。模型的单个输出通过另外 q 个单元抽头延迟线记忆反馈到输入。两个抽头延迟线记忆的内容被用于反馈到多层感知器的输入。模型输入的当前值用 $u(n)$ 代表，相对应的输出用 $y(n+1)$ 表示；也就是输出领先输入一个时间单位。因此应用到多层感知器输入层的信号向量的数据窗口数据如下：

- 现在和过去的输入值，即 $u(n), u(n-1), \cdots, u(n-q+1)$，表示来自网络外部的输入。
- 输出的延迟值，即 $y(n), y(n-1), \cdots, y(n-q+1)$，在此基础上模型输出 $y(n+1)$ 进行回归。

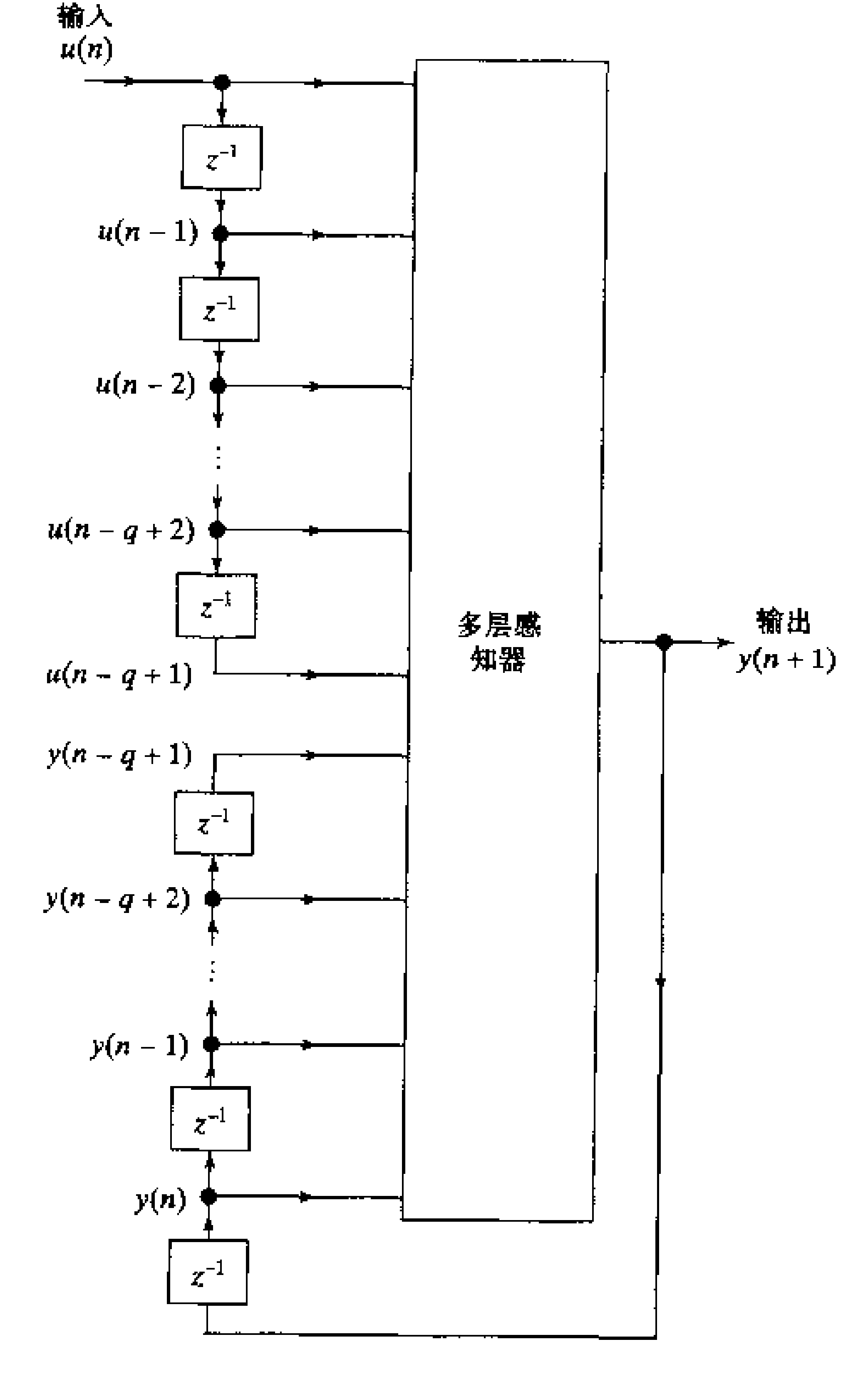


图15-1　有外部输入的非线性自回归(NARX)模型

因此图15-1的递归网络称为*有外部输入的非线性自回归模型*(nonlinear autoregressive with exogenous inputs model, NARX)[2]。NARX的动态行为由

$$y(n+1) = F(y(n), \cdots, y(n-q+1), u(n), \cdots, u(n-q+1)) \tag{15.1}$$

734 描述，其中 F 是它的自变量的一个非线性函数。注意在图15-1中，已经假

设两个延迟线记忆有同样大小的 q；它们一般是不同的。NARX 模型将在 15.4 节详细探究。

状态空间模型

图 15-2 表示另一种通用的递归网络的框图，称为*状态空间模型*。隐藏神经元定义网络的状态。隐藏层的输出通过一个单元延迟模块反馈回输入。输入层为反馈节点和源节点的联合。网络是通过源节点和外部连接的。用于将隐藏层输出反馈回输入层的延迟单元的数目决定了模型的阶数。$m \times 1$ 维的向量 $\mathbf{u}(n)$代表输入，$q \times 1$ 向量 $\mathbf{x}(n)$代表隐藏层在 n 时刻的输出向量。我们可以用下列两个联立方程组描述在图 15-2 中的模型的动态行为：

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n), \mathbf{u}(n)) \tag{15.2}$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{x}(n) \tag{15.3}$$

这里 $\mathbf{f}(\cdot,\cdot)$是一个刻划隐藏层特征的非线性函数，$\mathbf{C}$ 是代表输出层特征的突触权值矩阵。隐藏层是非线性的，但输出层是线性的。

图 15-2 的递归网络包括几个特殊的递归结构作为其特例。例如，Elman(1990)描述过的在图 15-3 所示的*简单递归网络*(simple recurrent network, SRN)。Elman 网络结构和图 15-2 所示结构有相似之处，除了输出层可以是非线性的和省略了输出的单元延迟模块。

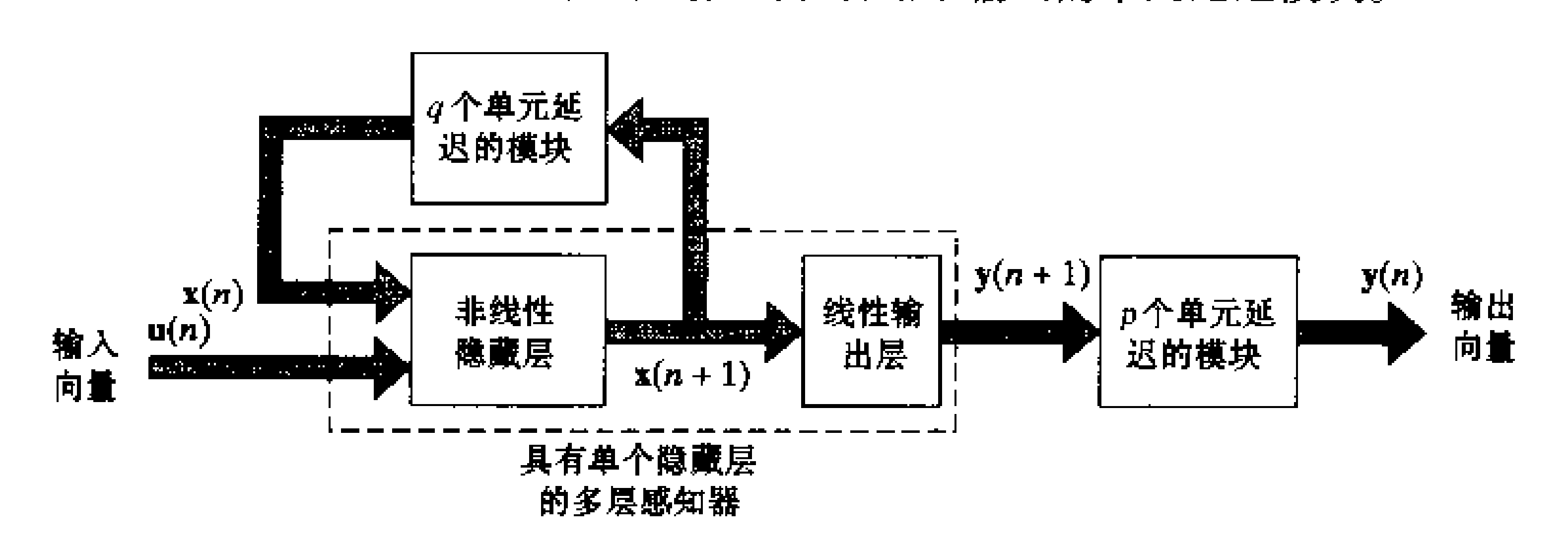


图 15-2 状态空间模型

735

Elman 网络包含从隐藏层神经元到由单元延迟组成的*背景单元*(context unit)层之间的递归连接。这些背景单元存储隐藏神经元对应一个时间步的输出，接着反馈回输入层。因此隐藏神经元具有它们以前激活的记录，这使得网络可以进行通过时间扩展的学习任务。隐藏神经元也馈给输出神经元，输出神经元给出在外部激励作用下网络的响应。由于隐藏神经元反馈的特性，这些神经元在多时间步内通过网络继续循环信息，从而发现时间的抽象表示。因此简单递归网络不仅仅是纪录过去数据的纪录带。

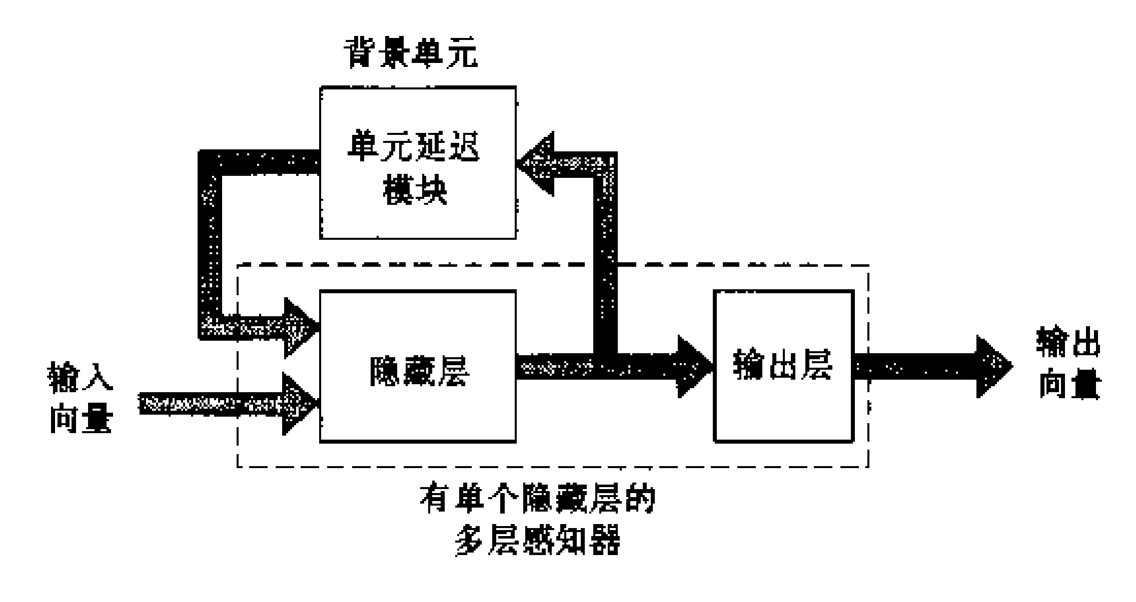


图 15-3 简单递归网络(SRN)

Elman(1990)讨论利用图 15-3 所示的简单递归网络在连续音素流中发现单词的边界，而不需任何内部表示性约束。递归网络的输入代表当前的音素。输出代表网络对序列中下一个

音符的最佳猜测。背景单元的作用是给网络提供动态记忆以便能够对包含在一系列的音素中的信息进行编码，这是和预测有关的。

递归多层感知器

736

第三种递归结构是一种*递归多层感知器*(recurrent multilayer perceptron, RMLP)(Puskorius et al., 1996)。它有一个或多个隐藏层，基本上因为同样的原因，静态多层感知器比那些使用单个隐藏层的感知器更有效和节约。RMLP 的每一个计算层对它的邻近层有一个反馈，如图 15-4 所示，此时 RMLP 有两个隐藏层[3]。

向量 $\mathbf{x}_{\mathrm{I}}(n)$代表第一个隐藏层的输出，$\mathbf{x}_{\mathrm{II}}(n)$ 代表第二个隐藏层的输出，以此类推。向量 $\mathbf{x}_o(n)$代表输出层的输出。那么，RMLP 通常对输入向量 $\mathbf{u}(n)$的响应的动态行为可用如下联立方程组描述：

$$\begin{aligned}\mathbf{x}_{\mathrm{I}}(n+1) &= \boldsymbol{\varphi}_{\mathrm{I}}(\mathbf{x}_{\mathrm{I}}(n),\mathbf{u}(n))\\ \mathbf{x}_{\mathrm{II}}(n+1) &= \boldsymbol{\varphi}_{\mathrm{II}}(\mathbf{x}_{\mathrm{II}}(n),\mathbf{x}_{\mathrm{I}}(n+1))\\ &\vdots\\ \mathbf{x}_o(n+1) &= \boldsymbol{\varphi}_o(\mathbf{x}_o(n),\mathbf{x}_K(n+1))\end{aligned} \tag{15.4}$$

其中 $\boldsymbol{\varphi}_{\mathrm{I}}(\cdot,\cdot)$，$\boldsymbol{\varphi}_{\mathrm{II}}(\cdot,\cdot)$，…，$\boldsymbol{\varphi}_o(\cdot,\cdot)$分别表示代表 RMLP 第一个隐藏层、第二个隐藏层、……和输出层的激活函数；K 表示网络中隐藏层的数目。

这里描述的 RMLP 包括图 15-3 的 Elman 网络和图 15-2 的状态空间模型，因为 RMLP 的输出层或任何隐藏层没有限定其激活函数的具体形式。

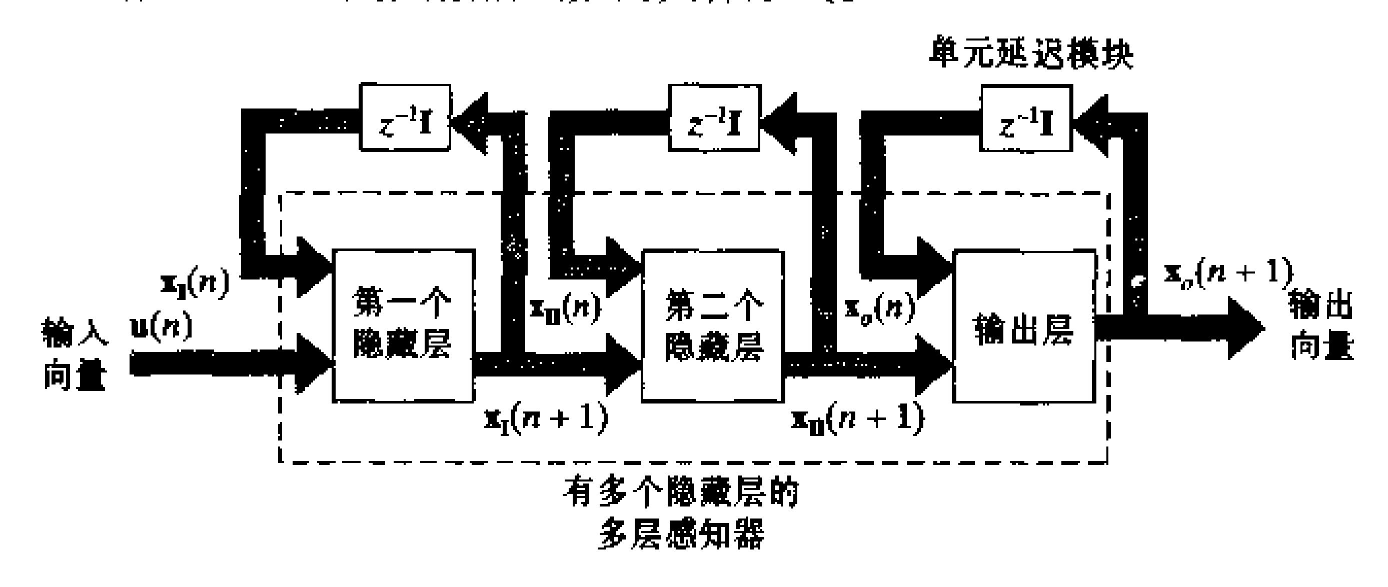


图 15-4 回归多层感知器

二阶网络

在描述图 15-2 的状态空间模型中，我们用"阶"来表示隐藏神经元的数目，其输出通过单元延迟模块反馈回输入层。

但是，术语"阶"有时用来表示如何定义神经元的诱导局部域的方法。例如，一个多层感知器神经元 k 的诱导局部域 v_k 定义为

$$v_k = \sum_j w_{a,kj}x_j + \sum_i w_{b,ki}u_i \tag{15.5}$$

其中 x_j 源于隐藏层神经元j 的反馈信号，u_i 是输入层应用于节点 i 的源信号；w 表示网络中对应的突触权值。将式(15.5)所描述的神经元称为*一阶神经元*。但是，有时诱导局部域 v_k 由乘法组成，表示为

$$v_k = \sum_i \sum_j w_{kji} x_i u_j \tag{15.6}$$

我们称这里的神经元为二阶神经元。二阶神经元 k 用了单一的权值 w_{kji}，它和输入节点 i，j 连接起来。

二阶神经元组成基本的二阶递归网络(Giles et al.,1990)，它的一个例子如图 15-5 所示。网络接受按时间顺序的输入序列，并且按如下两个式子定义的动力学演化：

$$v_k(n) = b_k + \sum_i \sum_j w_{kji} x_i(n) u_j(n) \tag{15.7}$$

737

$$x_k(n+1) = \varphi(v_k(n)) = \frac{1}{1+\exp(-v_k(n))} \tag{15.8}$$

其中 $v_k(n)$为隐藏神经元 k 的诱导局部域，b_k 为相关联的偏置，$x_k(n)$为神经元 k 的状态(输出)，$u_j(n)$是应用于源节点 j 的输入，w_{kji}为二阶神经元k 的权值。

图 15-5 所示的二阶递归网络的一个特点是乘积 $x_j(n)u_j(n)$代表一对{状态，输入}，一个正的权值 w_{kji}表示从{状态，输入}到{下一个状态}的状态转移的出现，而权值为负表示没有转移出现。状态转移描述如下：

$$\delta(x_i, u_j) = x_k \tag{15.9}$$

738

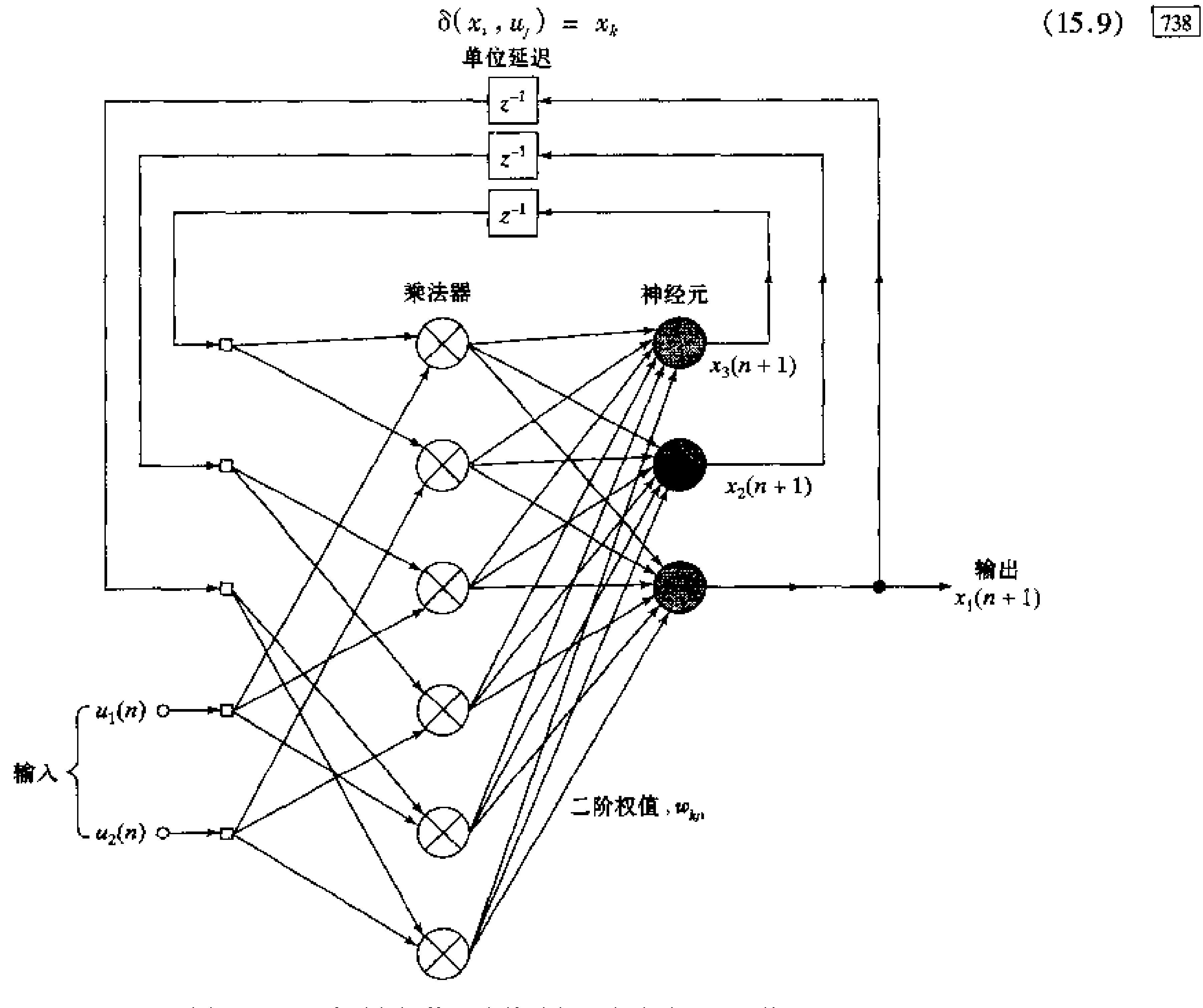


图 15-5 二阶递归网络；为简单起见省略神经元的偏置连接
网络有 2 个输入和 3 个状态神经元，因此需要 3×2=6 个乘法器

根据这种关系，二阶网络可以用来表示和学习确定性有限状态自动机[4]（deterministic finite-state automata，DFA），DFA 是一个有确定状态数目的信息处理装置。在15.5节可以发现更多关于神经网络和自动机关系的细节。

本节讨论的递归网络的体系结构强调利用全局反馈。如在简介中所提到的，递归网络也可能只有一个局部反馈。对后面这种递归网络性质的概述在 Tsoi and Back(1994)中提到；也可参见习题15.7。

15.3 状态空间模型

在动态系统的数学描述上，状态的概念起着重要的作用。动态系统的状态形式地定义为*一些数量的集合，它概括为了惟一地描述系统将来行为所必需的系统过去行为的全部信息，除了用于输入(激励)产生的外部效果之外*。$q\times 1$ 向量 $\mathbf{x}(n)$ 表示非线性离散时间系统的状态。$m\times 1$ 向量 $\mathbf{u}(n)$ 表示用于系统的输入，$p\times 1$ 向量 $\mathbf{y}(n)$ 表示相应的输出。使用数学语言，假设*无噪声*，系统的动态行为用非线性方程组

$$\mathbf{x}(n+1)=\boldsymbol{\varphi}(\mathbf{W}_a\mathbf{x}(n)+\mathbf{W}_b\mathbf{u}(n)) \tag{15.10}$$

$$\mathbf{y}(n)=\mathbf{C}\,\mathbf{x}(n) \tag{15.11}$$

描述，其中 $\mathbf{W}_a$ 是 $q\times q$ 矩阵，$\mathbf{W}_b$ 是 $q\times(m+1)$ 矩阵，$\mathbf{C}$ 是 $p\times q$ 矩阵；$\boldsymbol{\varphi}$: $\mathbb{R}^q\rightarrow\mathbb{R}^q$ 是对角映射，由

739

$$\boldsymbol{\varphi}:\begin{bmatrix}x_1\\x_2\\\vdots\\x_q\end{bmatrix}\rightarrow\begin{bmatrix}\varphi(x_1)\\\varphi(x_2)\\\vdots\\\varphi(x_q)\end{bmatrix} \tag{15.12}$$

描述，表示某种无记忆的分量非线性 φ: $\mathbb{R}\rightarrow\mathbb{R}$。空间 $\mathbb{R}^m$，$\mathbb{R}^q$ 和 $\mathbb{R}^p$ 分别称为*输入空间*、*状态空间*和*输出空间*。状态空间的大小(即 q)是系统的阶。因此图15-2的状态空间模型是 *m 输入、p 输出的 q 阶回归模型*。式(15.10)是模型的*过程方程*，式(15.11)是*度量方程*。过程方程(15.10)是式(15.2)的特殊形式。

建立在使用静态多层感知器和两个延迟线记忆基础上的图15-2的递归网络提供一种实现式(15.10)和(15.12)非线性反馈系统的方法。注意图15-2，*在多层感知器的神经元中，只有那些通过延迟将其输出反馈到输入层的神经元与确定递归网络的状态有关*。因此这就把输出层的神经元排除在状态的定义之外。

对于矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{C}$ 的解释，以及对非线性函数 $\varphi(\cdot)$，可以作如下陈述：

- 矩阵 $\mathbf{W}_a$ 代表隐藏层的 q 个神经元连接到输入层的反馈节点的突触权值。矩阵 $\mathbf{W}_b$ 代表连接到输入层源节点的这些隐藏神经元的突触权值。这里假设隐藏层神经元的偏置被包括在权值矩阵 $\mathbf{W}_b$ 中。
- 矩阵 $\mathbf{C}$ 代表输出层中连接到隐含神经元的 p 个线性神经元的突触权值。这里假设输出神经元的偏置被包括在权值矩阵 $\mathbf{C}$ 中。
- 非线性函数 $\varphi(\cdot)$ 代表隐藏神经元的 sigmoid 激活函数。激活函数通常具有双曲正切的形式

$$\varphi(x)=\tanh(x)=\frac{1-e^{-2x}}{1+e^{-2x}} \tag{15.13}$$

或 logistic 函数的形式

$$\varphi(x)=\frac{1}{1+e^{-x}} \tag{15.14}$$

式(15.10)和(15.11)描述的状态空间模型递归网络的一个重要性质，是它能逼近一类很大范围的非线性动态系统。但是，这种逼近只在一个状态空间的紧子集和有限的时间区间的情况下有效，所以感兴趣的动态特征并没有反映出来(Sontag,1992)。

例 15.1 为了表示矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{C}$ 的组成，考虑图 15-6 所示的完全连接递归网络，其中反馈路径来自隐藏神经元。在这个例中，$m=2$，$q=3$，$p=1$。矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 定义如下：

$$\mathbf{W}_a=\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix},\qquad \mathbf{W}_b=\begin{bmatrix} b_1 & w_{14} & w_{15} \\ b_2 & w_{24} & w_{25} \\ b_3 & w_{34} & w_{35} \end{bmatrix}$$

其中矩阵 $\mathbf{W}_b$ 的第一列由 b_1，b_2，b_3 组成，分别代表神经元 1，2，3 的偏置项。矩阵 $\mathbf{C}$ 是一个行向量，定义为 $\mathbf{C}=[1,0,0]$。 ■ 740

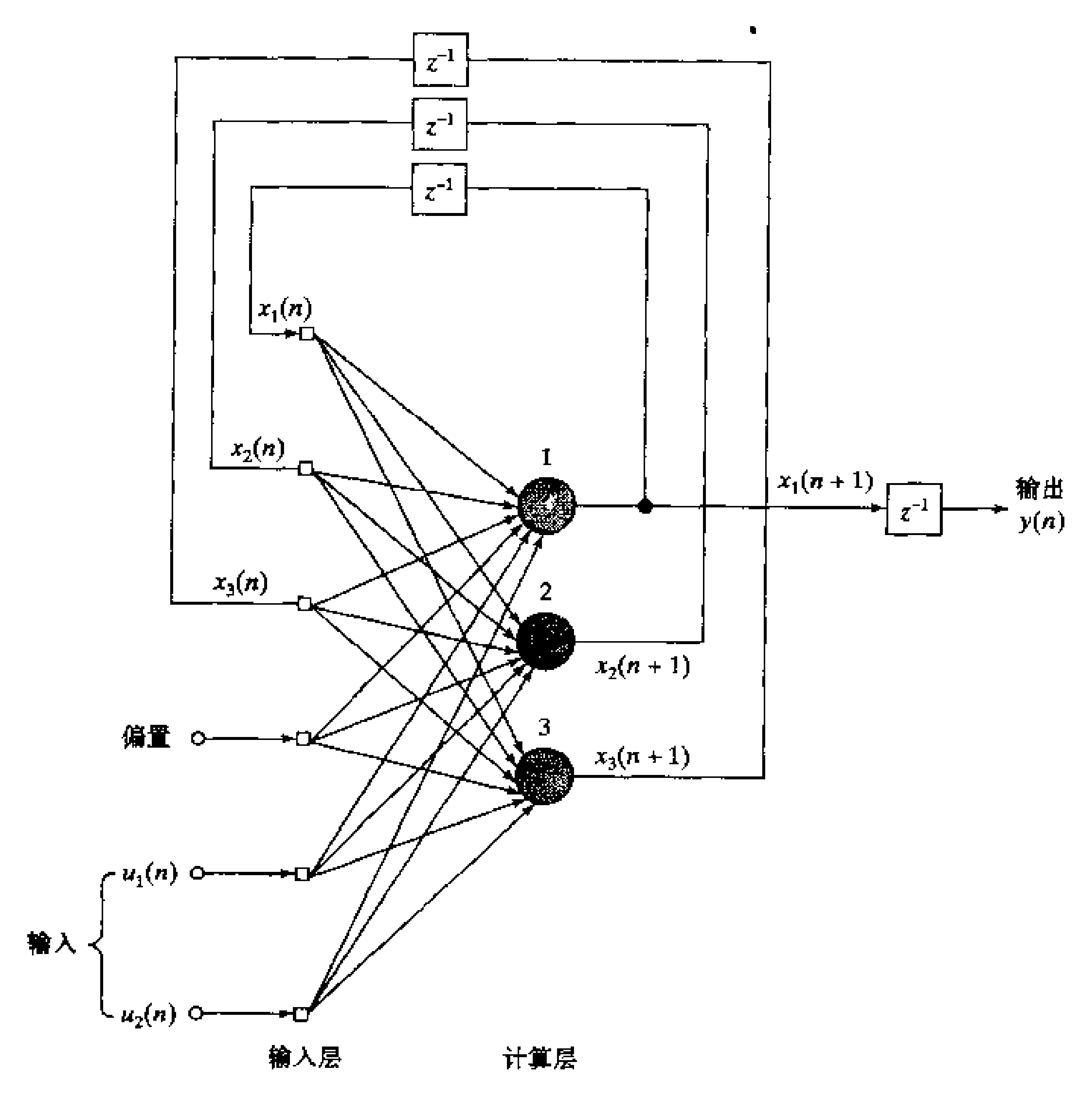


图 15-6 有两个输入、两个隐藏神经元和一个输出神经元的完全连接递归网络

可控性和可观察性

研究系统理论时，稳定性、可控性和可观察性以各自根本的方式为突出特征。本节讨论

可控性和可观察性，因为它们经常被放在一起处理。稳定性在前一章已讨论，不再详述。

741

前面已提到过，许多递归网络能用图15-2所示的状态空间模型表示，其中状态定义为通过一系列延迟单元反馈回输入层的隐藏层输出。在此背景下，知道递归网络是否可控和可观察是很重要的。可控性是指我们能否控制递归网络的动态行为。可观察性是指我们能否观察到应用于递归网络的控制结果。从这种意义来说，可观察性是可控性的对偶。

说递归网络是*可控的*，是指在有限时间步内，初始状态可以控制到任意想达到的状态；输出与这个定义无关。说递归网络是*可观察的*，是指在有限的输入/输出度量中网络的状态可以确定。对递归网络可控性和可观察性的精确的论述不在本书的讨论范围[5]。我们将自己限制在可控性和可观察性的*局部*形式。局部是指将这些概念应用于网络平衡状态邻域的意义下(Levin and Narendra, 1993)。

如果对于输入 $\mathbf{u}$，它满足条件

$$\bar{\mathbf{x}} = \boldsymbol{\varphi}(\mathbf{A}\bar{\mathbf{x}} + \mathbf{B}\bar{\mathbf{u}}) \tag{15.15}$$

就说状态 $\bar{\mathbf{x}}$ 是方程(15.10)的一个平衡状态。不失一般性，令 $\bar{\mathbf{x}} = \mathbf{0}$ 和 $\bar{\mathbf{u}} = \mathbf{0}$。那么平衡状态由

$$\mathbf{0} = \boldsymbol{\varphi}(\mathbf{0})$$

描述。换句话说，原点$(\mathbf{0}, \mathbf{0})$代表平衡点。

同样不失一般性，我们可以限制到一个单输入、单输出(single input, single output, SISO)系统，简化我们的论述。可以把方程(15.10)和(15.11)分别改写为

$$\mathbf{x}(n+1) = \boldsymbol{\varphi}(\mathbf{W}_a\mathbf{x}(n) + \mathbf{w}_b u(n)) \tag{15.16}$$

$$y(n) = \mathbf{c}^T\mathbf{x}(n) \tag{15.17}$$

其中 $\mathbf{w}_b$ 和 $\mathbf{c}$ 都是 $q \times 1$ 列向量，$u(n)$是标量输入，$y(n)$为标量输出。由于 $\boldsymbol{\varphi}$ 对应于式(15.13)或式(15.14)的sigmoid函数是连续可微的，我们可以通过在平衡点 $\bar{\mathbf{x}} = \mathbf{0}$ 和 $\bar{\mathbf{u}} = \mathbf{0}$ 的附近把式(15.16)展开成Taylor级数而使其线性化，并保留一阶项，得到

$$\delta\mathbf{x}(n+1) = \boldsymbol{\varphi}'(\mathbf{0})\mathbf{W}_a\delta\mathbf{x}(n) + \boldsymbol{\varphi}'(\mathbf{0})\mathbf{w}_b\delta u(n) \tag{15.18}$$

其中 $\delta\mathbf{x}(n)$和 $\delta u(n)$是分别应用到状态和输入的小位移。$q \times q$ 矩阵 $\boldsymbol{\varphi}'(\mathbf{0})$是 $\boldsymbol{\varphi}(\mathbf{v})$在 $\mathbf{v} = \mathbf{0}$ 时对变量 $\mathbf{v}$ 的Jacobi行列式。我们可以描述线性化的系统如下：

$$\delta\mathbf{x}(n+1) = \mathbf{A}\delta\mathbf{x}(n) + \mathbf{b}\delta u(n) \tag{15.19}$$

$$\delta y(n) = \mathbf{c}^T\delta\mathbf{x}(n) \tag{15.20}$$

其中 $q \times q$ 矩阵 $\mathbf{A}$ 和 $q \times 1$ 列向量 $\mathbf{b}$ 分别定义如下：

$$\mathbf{A} = \boldsymbol{\varphi}'(0)\mathbf{W}_a \tag{15.21}$$

742

$$\mathbf{b} = \boldsymbol{\varphi}'(0)\mathbf{w}_b \tag{15.22}$$

状态方程(15.19)和(15.20)是标准的线性形式。因此我们可以利用线性动态系统的可控性和可观察性的众所周知的结果，它们是数学控制论的一个标准部分。

局部可控性

从线性化的方程(15.19)，重复迭代产生下列结果：

$\delta\mathbf{x}(n+1) = \mathbf{A}\delta\mathbf{x}(n) + \mathbf{b}\delta u(n)$

$\delta\mathbf{x}(n+2) = \mathbf{A}\delta\mathbf{x}(n+1) + \mathbf{b}\delta u(n+1)$

$\vdots$

$$\delta\mathbf{x}(n+q)=\mathbf{A}^{q}\mathbf{b}\delta\mathbf{x}(n)+\mathbf{A}^{q-1}\mathbf{b}\delta u(n+q-1)+\cdots+\mathbf{A}\mathbf{b}\delta u(n+1)+\mathbf{b}\delta u(n)$$

其中 q 是状态空间的维数。相应地，我们可以说(Levin and Narendra,1993)：

方程(15.19)表示的线性化系统是可控的，如果矩阵

$$\mathbf{M}_c=[\mathbf{A}^{q-1}\mathbf{b},\cdots,\mathbf{A}\mathbf{b},\mathbf{b}] \tag{15.23}$$

有秩 q，即满秩，因为这样线性化的过程方程(15.19)有惟一的解。

矩阵 $\mathbf{M}_c$ 称为线性系统的可控性矩阵。

设方程(15.16)和(15.17)描述的递归网络由一系列输入 $\mathbf{u}_q(n)$ 驱动，其定义为

$$\mathbf{u}_q(n)=[u(n),u(n+1),\cdots,u(n+q-1)]^T \tag{15.24}$$

因此可以考虑映射

$$\mathbf{G}(\mathbf{x}(n),\mathbf{u}_q(n))=(\mathbf{x}(n),\mathbf{x}(n+q)) \tag{15.25}$$

其中 $\mathbf{G}$: $\mathbb{R}^{2q}\rightarrow\mathbb{R}^{2q}$。在习题 15.4 证明：

- 状态 $\mathbf{x}(n+q)$ 是其过去值 $\mathbf{x}(n)$ 和输入 $u(n)$，$u(n+1)$，…，$u(n+q-1)$ 的嵌套非线性函数。
- $\mathbf{x}(n+q)$ 关于 $\mathbf{u}_q(n)$ 的 Jacobi 矩阵在原点的值等于式(15.23)的可控性矩阵 $\mathbf{M}_c$。

我们可以把映射 $\mathbf{G}$ 关于 $\mathbf{u}_q(n)$ 和 $\mathbf{x}(n)$ 的 Jacobi 矩阵在原点($\mathbf{0}$，$\mathbf{0}$)的值表示为

$$\mathbf{J}^{(c)}_{(0,0)}=\begin{bmatrix}\left(\dfrac{\partial\mathbf{x}(n)}{\partial\mathbf{x}(n)}\right)_{(0,0)} & \left(\dfrac{\partial\mathbf{x}(n+q)}{\partial\mathbf{x}(n)}\right)_{(0,0)}\\ \left(\dfrac{\partial\mathbf{x}(n)}{\partial\mathbf{u}_q(n)}\right)_{(0,0)} & \left(\dfrac{\partial\mathbf{x}(n+q)}{\partial\mathbf{u}_q(n)}\right)_{(0,0)}\end{bmatrix}=\begin{bmatrix}\mathbf{I} & \mathbf{X}\\ \mathbf{0} & \mathbf{M}_c\end{bmatrix} \tag{15.26}$$

743

其中 $\mathbf{I}$ 是单位矩阵，$\mathbf{0}$ 是零矩阵，项 $\mathbf{X}$ 是不感兴趣的部分。因为它的特殊形式，$\mathbf{J}^{(c)}_{(0,0)}$ 的行列式等于单位矩阵 $\mathbf{I}$ 的行列式(等于 1)和可控性矩阵 $\mathbf{M}_c$ 的行列式乘积。如果 $\mathbf{M}_c$ 是满秩矩阵，那么 $\mathbf{J}^{(c)}_{(0,0)}$ 也是满秩的。

为了继续处理，我们需要引用*反函数定理*，它可以陈述如下(Vidyasagar,1993)：

考虑映射 $\mathbf{f}$: $\mathbb{R}^q\rightarrow\mathbb{R}^q$，假设映射 $\mathbf{f}$ 的每一个分量对于它的变量在平衡点 $\mathbf{x}_0\in\mathbb{R}^q$ 都是可微的，并令 $\mathbf{y}_0=\mathbf{f}(\mathbf{x}_0)$。那么存在开集 $\mathcal{U}\subseteq\mathbb{R}^q$ 包含 $\mathbf{x}_0$ 及 $\mathcal{V}\subseteq\mathbb{R}^q$ 包含 $\mathbf{y}_0$，使得 $\mathbf{f}$ 为 $\mathcal{U}$ 到 $\mathcal{V}$ 上的微分同胚。如果 $\mathbf{f}$ 还是光滑的，那么逆映射 $\mathbf{f}^{-1}$: $\mathbb{R}^q\rightarrow\mathbb{R}^q$ 也是光滑的，即 $\mathbf{f}$ 是光滑微分同胚。

映射 $\mathbf{f}$: $\mathcal{U}\rightarrow\mathcal{V}$ 如果满足下列条件，则说它是 $\mathcal{U}$ 到 $\mathcal{V}$ 上的微分同胚：

1. $\mathbf{f}(\mathcal{U})=\mathcal{V}$。
2. 映射 $\mathbf{f}$: $\mathcal{U}\rightarrow\mathcal{V}$ 是一对一的(即可逆的)。
3. 逆映射 $\mathbf{f}^{-1}$: $\mathcal{V}\rightarrow\mathcal{U}$ 的每个分量关于它的变量是连续可微的。

回到可控性的问题，我们将对式(15.25)定义的映射验证满足反函数定理中的 $\mathbf{f}(\mathcal{U})=\mathcal{V}$ 条件。应用反函数定理，如果可控性矩阵 $\mathbf{M}_c$ 的秩为 q，可以说局部存在一个反映射，定义为

$$(\mathbf{x}(n),\mathbf{x}(n+q))=\mathbf{G}^{-1}(\mathbf{x}(n),\mathbf{u}_q(n)) \tag{15.27}$$

式(15.27)实际上指出存在一个输入序列能局部驱动网络在 q 个时间步中从状态 $\mathbf{x}(n)$ 到 $\mathbf{x}(n+q)$。所以，我们可以正式陈述*局部可控性定理*如下：

假定递归网络由式(15.16)和(15.17)定义，它在原点(即平衡点)附近的线性化方程由(15.19)和(15.20)定义。如果线性化系统是可控的，则递归网络是在原点附近是局部可控的。

局部可观察性

重复使用线性化的方程(15.19)和(15.20)，可得

$$\begin{aligned}\delta y(n) &= \mathbf{c}^T\delta\mathbf{x}(n)\\ \delta y(n+1) &= \mathbf{c}^T\delta\mathbf{x}(n+1)\\ &= \mathbf{c}^T\mathbf{A}\delta\mathbf{x}(n) + \mathbf{c}^T\mathbf{b}\delta u(n)\\ &\vdots\\ \delta y(n+q-1) &= \mathbf{c}^T\mathbf{A}^{q-1}\delta\mathbf{x}(n) + \mathbf{c}^T\mathbf{A}^{q-2}\mathbf{b}\delta u(n) + \cdots + \mathbf{c}^T\mathbf{A}\mathbf{b}\delta u(n+q-3)\\ &\quad + \mathbf{c}^T\mathbf{b}\delta u(n+q-2)\end{aligned}$$

744

其中 q 是状态空间的维数。所以，我们可以陈述(Levin and Narendra, 1993)：

方程(15.19)和(15.20)描述的线性化系统是可观察的，如果矩阵

$$\mathbf{M}_o = [\mathbf{c}, \mathbf{c}\mathbf{A}^T, \cdots, \mathbf{c}(\mathbf{A}^T)^{q-1}] \tag{15.28}$$

的秩为 q，即满秩。

矩阵 $\mathbf{M}_o$ 称为线性系统的可观察性矩阵。

令用于驱动由式 (15.19)和(15.20)描述的递归网络的一系列输入定义如下：

$$\mathbf{u}_{q-1}(n) = [u(n), u(n+1), \cdots, u(n+q-2)]^T \tag{15.29}$$

相应地，令

$$\mathbf{y}_q(n) = [y(n), y(n+1), \cdots, y(n+q-1)]^T \tag{15.30}$$

代表由初始状态 $\mathbf{x}(n)$和输入序列 $\mathbf{u}_{q-1}(n)$产生的输出向量。那么我们可以考虑映射

$$\mathbf{H}(\mathbf{u}_{q-1}(n), \mathbf{x}(n)) = (\mathbf{u}_{q-1}(n), \mathbf{y}_q(n)) \tag{15.31}$$

其中 $\mathbf{H}$: $\mathbb{R}^{2q-1}\to\mathbb{R}^{2q-1}$。在习题 15.5 中证明 $\mathbf{y}_q(n)$对 $\mathbf{x}(n)$的 Jacobi 矩阵在原点的值等于式(15.28)的可观察矩阵 $\mathbf{M}_o$。因此 $\mathbf{H}$ 关于 $\mathbf{u}_{q-1}(n)$和 $\mathbf{x}(n)$的 Jacobi 矩阵在原点(0，0)的值可表示为

$$\mathbf{J}_{(0,0)}^{(o)} = \begin{bmatrix} \left(\dfrac{\partial \mathbf{u}_{q-1}(n)}{\partial \mathbf{u}_{q-1}(n)}\right)_{(0,0)} & \left(\dfrac{\partial \mathbf{y}_q(n)}{\partial \mathbf{u}_{q-1}(n)}\right)_{(0,0)} \\ \left(\dfrac{\partial \mathbf{u}_{q-1}(n)}{\partial \mathbf{x}(n)}\right)_{(0,0)} & \left(\dfrac{\partial \mathbf{y}_q(n)}{\partial \mathbf{x}(n)}\right)_{(0,0)} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{X} \\ \mathbf{0} & \mathbf{M}_o \end{bmatrix} \tag{15.32}$$

其中 $\mathbf{X}$ 同样为不感兴趣的部分。$\mathbf{J}_{(0,0)}^{(o)}$的行列式等于单位矩阵 $\mathbf{I}$ 的行列式(等于 1)和矩阵 $\mathbf{M}_o$ 的行列式的乘积。如果 $\mathbf{M}_o$ 是满秩，那么 $\mathbf{J}_{(0,0)}^{(o)}$ 也是。引用反函数定理，可以说如果线性化系统的可观察性矩阵 $\mathbf{M}_o$ 是满秩的，则存在一个逆映射，定义为

745

$$(\mathbf{u}_{q-1}(n), \mathbf{x}(n)) = \mathbf{H}^{-1}(\mathbf{u}_{q-1}(n), \mathbf{y}_q(n)) \tag{15.33}$$

实际上，这个等式表明在原点的局部邻域，$\mathbf{x}(n)$是 $\mathbf{u}_{q-1}(n)$和 $\mathbf{y}_q(n)$的非线性函数，非线性函数是递归网络的观察器。因此局部可观察性定理可正式地陈述如下(Levin and Narendra, 1993)：

由式(15.16)和(15.17)所定义的递归网络，令它在原点(即平衡点)附近线性化的形式由式(15.19)和(15.20)所定义。如果线性系统是可观察的，则递归网络在原点附近是可观察的。

例 15.2 考虑具有矩阵 $\mathbf{A}=a\mathbf{I}$ 的状态空间模型，这里 a 是标量，$\mathbf{I}$ 是单位矩阵。式(15.23)的可控性矩阵 $\mathbf{M}_c$ 简化为

$$\mathbf{M}_c = a[\mathbf{b},\cdots,\mathbf{b},\mathbf{b}]$$

矩阵的秩是 1。因此，具有矩阵 $\mathbf{A}$ 的值的线性化系统是不可控的。

在式(15.28)中置 $\mathbf{A}=a\mathbf{I}$，得到可观察性矩阵

$$\mathbf{M}_o = a[\mathbf{c},\mathbf{c},\cdots,\mathbf{c}]$$

它的秩也为 1。这个线性系统也是不可观察的。 ■

15.4 有外部输入的非线性自回归模型

考虑单输入单输出的递归网络，其行为由状态方程组(15.16)和(15.17)描述。给定这种状态模型，希望将它修改为一个输入－输出模型，作为代表递归网络的一个等价表示。

利用式(15.16)和(15.17)，输出 $y(n+q)$ 可以用状态 $\mathbf{x}(n)$ 和输入向量 $\mathbf{u}_q(n)$ 表示为(参看习题 15.8)

$$y(n+q) = \Phi(\mathbf{x}(n),\mathbf{u}_q(n)) \tag{15.34}$$

其中 q 是状态空间的维数，Φ：$\mathbb{R}^{2q}\to\mathbb{R}$。假设递归网络为可观察的，可以用局部可观察性定理得到

$$\mathbf{x}(n) = \Psi(\mathbf{y}_q(n),\mathbf{u}_{q-1}(n)) \tag{15.35}$$

其中映射 Ψ：$\mathbb{R}^{2q-1}\to\mathbb{R}^{q}$。将式(15.35)代入(15.34)，得到

$$\begin{aligned} y(n+q) &= \Phi(\Psi(\mathbf{y}_q(n),\mathbf{u}_{q-1}(n)),\mathbf{u}_q(n)) \\ &= F(\mathbf{y}_q(n),\mathbf{u}_q(n)) \end{aligned} \tag{15.36}$$

其中 $\mathbf{u}_{q-1}(n)$ 包含在 $\mathbf{u}_q(n)$ 的最前面的 $q-1$ 个元素里，非线性映射 F：$\mathbb{R}^{2q}\to\mathbb{R}$ 和 Φ，Ψ 有关。用式(15.30)和(15.29)给出的 $\mathbf{y}_q(n)$ 和 $\mathbf{u}_q(n)$ 定义，可以将式(15.36)扩展为

$$y(n+q) = F(y(n+q-1),\cdots,y(n),u(n+q-1),\cdots u(n))$$

用 $n-q+1$ 代替 n，可以得到

$$y(n+1) = F(y(n),\cdots,y(n-q+1),u(n),\cdots,u(n-q+1)) \tag{15.37}$$

[746]

必须指出，对于这个非线性映射 F：$\mathbb{R}^{2q}\to\mathbb{R}$，只有当现在的输出 $y(n+1)$ 由过去值 $y(n)$，$\cdots$，$y(n-q+1)$ 以及现在和过去的输入 $u(n)$，$\cdots$，$u(n-q+1)$ 所惟一决定，这个映射才是存在的。因为这个输入－输出表示等价于方程组(15.16)和(15.17)的状态模型，因此递归网络必须是可观察的。等价的实际含义是图 15-1 的 NARX 模型，它的全局反馈限制在输出神经元，实际上它是能够模拟图 15-2 的完全回归状态空间模型(假设 $m=1$，$p=1$)，并且它们的输入－输出行为没有差别。

例 15.3 再考虑图 15-6 描述的完全连接递归网络。对于我们目前的讨论，假设其中一个输入，比如说 $u_2(n)$，削减为 0，这样我们有一个单输入、单输出的网络。如果网络是局部可观察的，可以用图 15-7 的 NARX 模型代替完全连接网络。虽然 NARX 模型仅有产生于输出神经元的有限反馈这种情况，而图 15-6 的完全连接递归网络的多层感知器周围的反馈产生于三个隐藏/输出神经元，但是这种等价性还是成立的。 ■

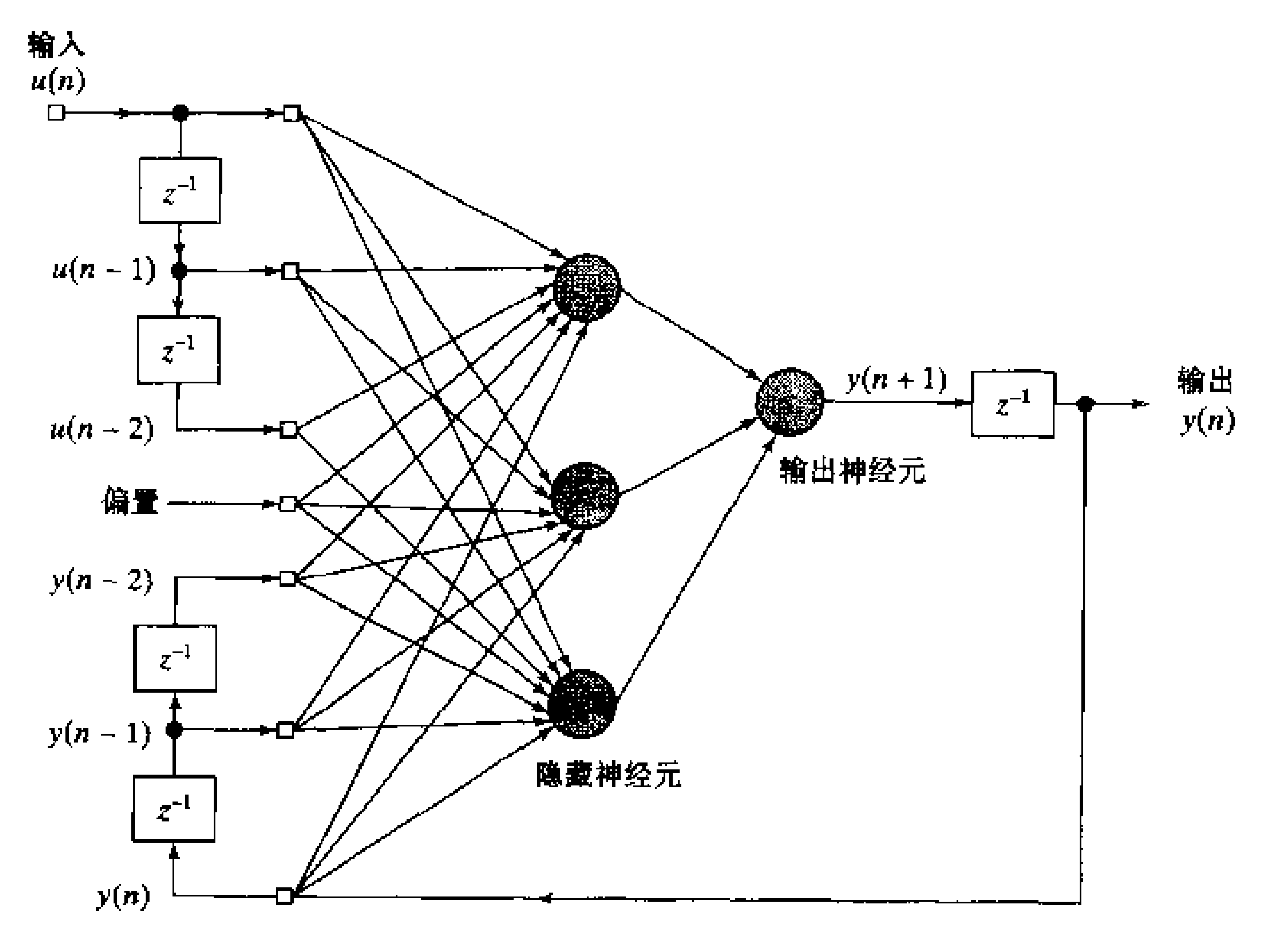


图 15-7 具有3个隐藏神经元的 NARX 网络

15.5 递归网络的计算能力

如图 15-2 所示的状态空间模型和图 15-1 的 NARX 模型，递归网络具有模拟有限状态自动机的固有能力。自动机表示像计算机一样的信息处理设备的抽象。实际上自动机和神经网络有久远的渊源[6]。Minsky 在他 1967 年的书(p.55)有如下重要的说明：

每一有限状态机等价于某神经网络，并且可以由它模拟。也就是说，给定一有限状态机 $\mathcal{M}$，可以建立一个神经网络 $\mathcal{N}^{\mathcal{M}}$，若将它看做一个黑箱机器，则其行为酷似 $\mathcal{M}$。

递归网络的早期工作用硬的阈值逻辑作为神经元的激活函数而不用软的 sigmoid 函数。

也许是 Cleeremans(1989)第一个报道了展示递归网络能否学会由小型有限状态语法所包含的例外(偶发性)的试验。特别地，由语法导出的字符串赋给简单递归网络(图 15-3)，需要它在每一步预测下一字母。预测是上下文相关的，因为每一个在语法中出现两次的字母每次它的后继字母都不同。这表明网络能够在隐藏神经元中发展对应自动机(有限状态机)状态的内部表示。在 Kremer(1995)中给出正式的证明，表明简单递归网络有和任何有限状态机一样的计算能力。

在一般意义下，递归网络的计算能力体现在两个主要定理：

定理 I (Siegelmann and Sontag, 1991) *所有图灵机都可由建立在用 sigmoid 激活函数的神经元上的完全连接递归网络模拟。*

图灵机是 Turing(1936)发明的抽象计算工具。它由图 15-8 所示的三个功能块构成：(1)控制单元假设任何可能的有限状态之一；(2)线性带(假设在两个方向上是无限的)被划分成分离的方块，每个方块都可以存储一个单一的符号，这些符号是从一个有限的符号集合中取出的；(3)读写头沿着线性带移动，并从控制单元得到信息和把信息传送到控制单元(Fischler

and Firschein, 1987)。从给出的讨论足以说明图灵机是一个和任何强大的计算机具有一样功能和能力的抽象物。这个思想称为 Church-Turing 假设。

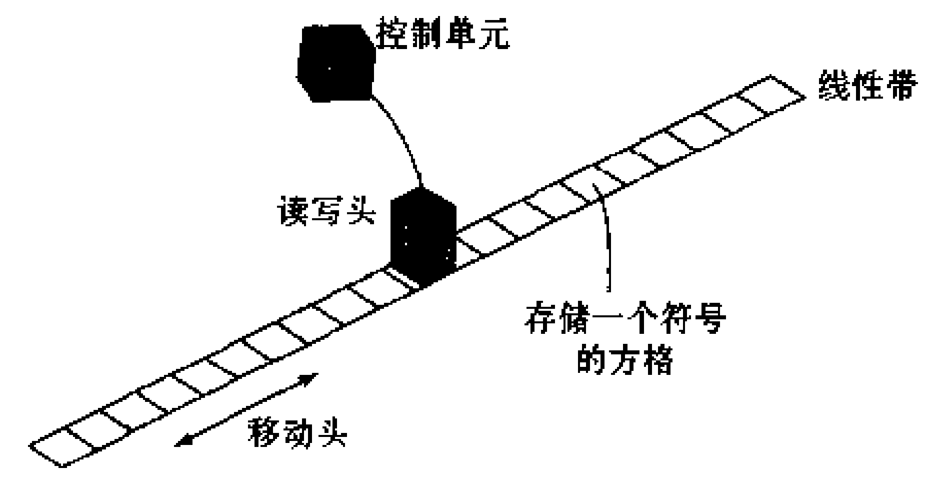


图 15-8 Turing 机

定理 II (siegelmann et al.,1997) 对于 NARX 网络，若具有一隐藏层单元，其激活函数为有界和单侧饱和的并且有一个线性输出神经元，那么不计线性延迟(linear slowdown)，它可以模拟用完全连接的具有有界且单侧饱和的激活函数的递归网络。

748

"线性延迟"是指如果一个完全连接的有 N 个神经元的递归网络在时间 T 内计算一个我们感兴趣的任务，那么等价的 NARX 网络所占用的总时间是$(N+1)T$。函数 $\varphi(\cdot)$如果满足下列条件则说它是有界且单边饱和的(bounded, one-sided saturatied, BOSS)函数：

1. 函数 $\varphi(\cdot)$值域有界；即 $a \leqslant \varphi(x) \leqslant b$，对于所有 $x \in \mathbb{R}$。
2. 函数 $\varphi(\cdot)$是左饱和的；即存在值 s 和 S，对于所有的 $x \leqslant s$，有 $\varphi(x)=S$。
3. 函数 $\varphi(\cdot)$是非常数的；即存在不相同的两个数 x_1 和 x_2，满足 $\varphi(x_1) \neq \varphi(x_2)$。

阈值(Heaviside)和分段线性函数满足 BOSS 条件。但是在严格意义上 sigmoid 函数不是一个 BOSS 函数，因为它不满足条件 2。但是做一个小的修改，它可以满足 BOSS 条件，即写成(在 logistic 函数的情况下)

$$\varphi(x)=\begin{cases}\dfrac{1}{1+\exp(-x)} & 对于\ x>s \\ 0 & 对于\ x \leqslant s\end{cases}$$

其中 $s \in \mathbb{R}$。实际上，在 $x \leqslant s$ 时 logistic 函数是截断的。

作为定理 I 和定理 II 的推论，我们可以得到(Giles, 1996)：

有一个隐藏层神经元且激活函数为 BOSS 函数及一个线性输出神经元的 NARX 网络是 Turing 等价的。

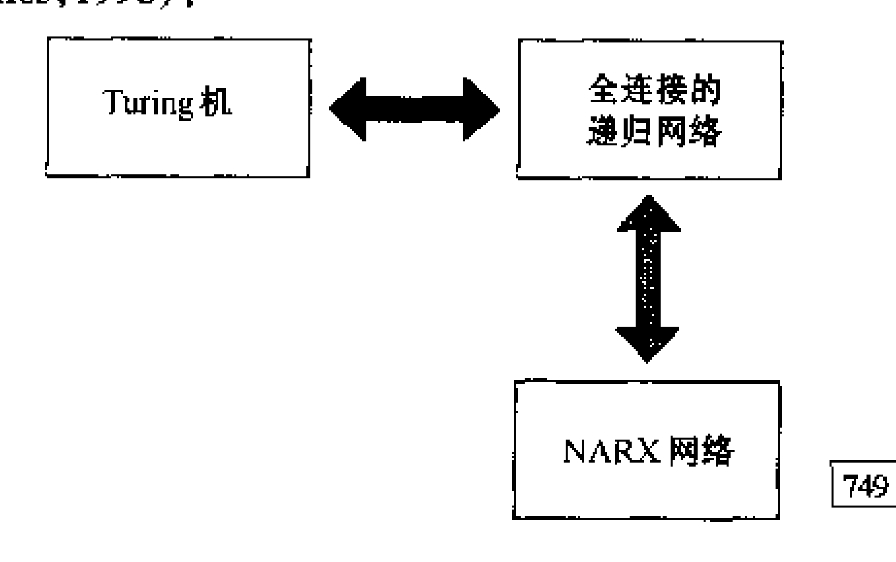


图 15-9 定理 I 和定理 II 及它们的推论的图解

图 15-9 给出定理 I 和定理 II 及这个推论的图解。但是，必须注意当网络体系结构受到限制时，递归网络的计算能力就不再成立，如同在 Sperduti(1997)描述的一样。在注释[7]中给出受限制的网络体系结构的参考文献。

749

15.6 学习算法

现在来研究递归网络的训练问题。第 4 章讨论过普通(静态)多层感知器的两种方式：集中方式和串行方式。在集中方式中，网络的敏感度是在调整网络的自由参数前针对整个训练集计算的。在串行方式中，参数的调整是在给出训练集的每一个模式的表示之后进行的。同样，有两个训练递归网络的方式如下：

1. *分回合(epochwise)的训练*。在给定的回合，递归网络从初始状态出发到达一个新的状态后停止，此时训练亦停止；然后对于下一个回合又重新设置一个新的初始状态。初始状态在每个训练时期并不总是一样的。重要的是对于新的回合的初始状态和网络在此前一个回合到达的状态不一样。例如，考虑用递归网络模拟有限状态机的运行，即一个设备可区分的内部配置(状态)在数量上是有限的。在这种条件下，有理由使用分回合的训练，因为我们有很大的可能性用递归网络去模拟机器中大量的不同的初始状态和不同的最终状态的集合。在递归网络的分回合训练中，"回合"与一般普通多层感知器中使用的意义不同。用现在的术语，递归网络的回合对应普通多层感知器的一个训练模式。

2. *连续训练*。训练的第二种方法适合于没有可用的重置状态和/或需要在线学习的情况。连续训练的显著特征是网络学习和被网络处理的信号处理同时进行。简单地说，学习过程永不停止。例如，考虑让递归网络去对一个非稳态过程如语音信号建模。在这种情况下，网络的连续运行不能提供方便的时刻以决定何时停止训练而重新开始用网络不同自由参数的值。

记住这两种训练的方式，在下面的两节中我们将描述递归网络的不同的学习算法，可概述如下：

- 在15.7节讨论的通过时间的反向传播(back-propagation-through-time)算法是在这样的前提下提出的，即递归网络的时序操作可以展开为一个多层感知器。这就为标准反向传播算法的应用铺平了道路。通过时间的反向传播算法可以用分回合的方式、连续方式或两种方式的组合来实现。
- 在15.8节讨论的实时递归学习算法是从方程(15.10)和(15.11)描述的状态空间模型导出的。

两种算法有很多共同点。首先它们都是基于梯度下降的方法，因此代价函数的瞬时值(基于平方误差准则)对网络的突触权值被最小化。第二，它们实现都很简单，但可能收敛很慢。第三，它们是相关的，因为通过时间的反向传播算法的信号流图的表示，能够由实时递归学习算法的一确定形式的信号流图的表示经*转置*而得到(Lefebvre,1991;Beaufays and Wan,1994)。

750 建立在梯度下降基础上的实时(连续)学习使用*最少可用信息*，即代价函数关于被调整参数向量的梯度的瞬时估值。可以通过利用Kalman的滤波理论加速学习过程，它更有效地利用包含在训练数据中的信息。在15.10节简单介绍解耦扩展的Kalman滤波器，通过它我们可以处理动态学习任务，而对用以梯度下降为基础的方法，这将会是非常困难的。在15.9节给出Kalman滤波器的简要回顾。注意解耦扩展的Kalman滤波器既可以应用于静态前馈网络，亦可应用于递归网络。

一些启发

在进行刚才提到的新学习算法的描述之前，我们罗列一些对于改进递归网络训练的启发，这些训练涉及梯度下降方法的使用(Giles,1996)：

- 训练样本应该按照字典顺序排序，最短的符号字符串首先提交给网络。
- 训练应该开始于一个小的训练样本集，尔后随着训练进行逐步增加样本。
- 只有当正在被网络处理的训练样本的绝对误差大于某一指定的标准时才应该更新网络的突触权值。
- 在训练过程中建议使用权值衰减；权值衰减作为复杂性正则化的一个粗略的形式，在

第4章讨论。

第一个启发有特别重要的意义。如果可以实现的话，它提供减轻在采用梯度下降方法训练递归网络时出现的消失梯度问题。这个问题在15.12节讨论。

15.7 通过时间的反向传播

用于训练一个递归网络的通过时间的反向传播(BPTT)算法是标准反向传播算法的扩展[18]。它可以通过将网络的时序操作展开成一个分层的前馈网络导出，它的拓扑结构在每个时间步增加一层。

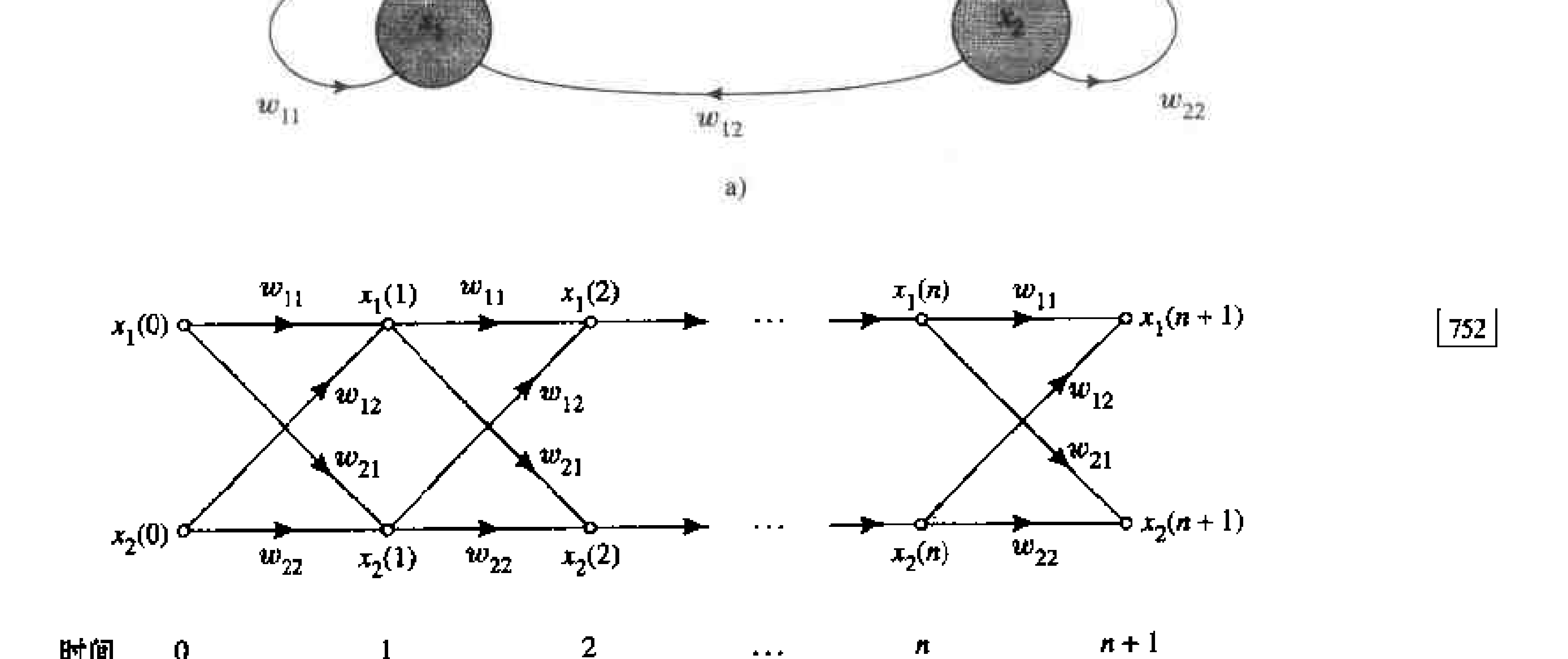
751

具体地，让$\mathscr{N}$表示需要学习时序任务的递归网络，从时间 n_0 开始一直到时间 n。$\mathscr{N}^*$ 表示对递归网络$\mathscr{N}$的时序操作进行展开所得的前馈网络。展开后的网络$\mathscr{N}^*$和初始网络$\mathscr{N}$的关系如下：

1. 对区间$(n_0, n]$内的每一个时间步，网络$\mathscr{N}^*$有一个包含 K 个神经元的层，K 是包含在网络$\mathscr{N}$中的神经元的数量。
2. 在网络$\mathscr{N}^*$的每一层有网络$\mathscr{N}$的每一个神经元的拷贝。
3. 对每一个时间步 $l \in [n_0, n]$，从网络$\mathscr{N}^*$中 l 层的神经元 i 到 $l+1$ 层的神经元 j 的突触连接，是在网络$\mathscr{N}$中从神经元 i 到神经元 j 的突触连接的拷贝。

这些要点在下面的例子中解释。

例 15.4 考虑图15-10a所示的两个神经元递归网络$\mathscr{N}$。为简化表示，省略单位延迟操作符 z^{-1}。这个操作符应该插入到图15-10a所示突触连接(包括自连接环)的每一步。通过一步一步地展开网络的时序操作，得到图15-10b的信号流图，其中起始时间 $n=0$。图15-10b代表分层的前馈网络$\mathscr{N}^*$，其中在每一步时序操作都有新的层加入。 ■



752

图 15-10

a)两个神经元递归网络$\mathscr{N}$的结构图 b)网络$\mathscr{N}$依时间展开的信号流图

依赖于使用分回合训练或使用连续(实时)训练，展开过程的应用导致通过时间的反向传播两个根本不同的实现。下面依次描述这两种递归学习方法。

分回合的通过时间的反向传播

将用于递归网络训练的数据集分割为独立的回合，每一回合表示一个感兴趣的时序模式。令 n_0 表示一个回合的开始时间，n_1 表示其结束时间。在这个回合里，可以定义代价函数

$$\mathscr{E}_{\text{total}}(n_0, n_1) = \frac{1}{2}\sum_{n=n_0}^{n_1}\sum_{j\in\mathscr{A}} e_j^2(n) \tag{15.38}$$

其中$\mathscr{A}$为网络中指定期望响应的那些神经元标号 j 的集合，$e_j(n)$是该神经元关于期望响应和计算出的实际输出之间的误差信号。我们希望计算网络的敏感度，即计算代价函数对网络突触权值的偏导数。为此，可以用通过时间的反向传播(back-propagation-through-time, BPTT)算法，这个算法建立在第4章讨论的标准反向传播学习集中方式的基础上。分回合的BPTT算法进行如下(Williams and Peng, 1990)：

- 首先，对时间区间(n_0, n_1)执行单纯的数据前向传播通过网络。保存完整的输入数据记录、网络状态(即网络的突触权值)以及期望响应。
- 对过去这条记录执行一个单纯的反向传播通过网络，计算局部梯度

$$\delta_j(n) = -\frac{\partial \mathscr{E}_{\text{total}}(n_0, n_1)}{\partial v_j(n)} \tag{15.39}$$

的值，对于所有的 $j\in\mathscr{A}$，$n_0 < n \leqslant n_1$。这个计算用公式

$$\delta_j(n) = \begin{cases}\varphi'(v_j(n))e_j(n) & \text{对于 } n = n_1\\ \varphi'(v_j(n))\left[e_j(n) + \sum_{k\in\mathscr{A}} w_{jk}\delta_k(n+1)\right] & \text{对于 } n_0 < n < n_1\end{cases} \tag{15.40}$$

进行，其中 $\varphi'(\cdot)$是激活函数对它的自变量的导数，$v_j(n)$是神经元 j 的诱导局部域。这里假设网络的所有神经元有同样的激活函数 $\varphi(\cdot)$。重复使用式(15.40)，从时刻 n_1 出发，向后一步一步进行直到时刻 n_0；此处涉及的步数与包含在这个回合内的步数相同。

- 一旦执行反向传播的计算回到 n_0+1 时，对神经元 j 的突触权值 w_{ji} 调整如下：

753

$$\Delta w_{ji} = -\eta\frac{\partial \mathscr{E}_{\text{total}}(n_0, n_1)}{\partial w_{ji}} = \eta\sum_{n=n_0+1}^{n_1}\delta_j(n)x_i(n-1) \tag{15.41}$$

其中 η 是学习率参数，$x_i(n-1)$是在时刻 $n-1$ 时作用于神经元 j 的第 i 个突触的输入。

比较刚才描述的分回合的BPTT的过程和标准反向传播学习的集中方式，可以看出它们根本的差别是前者在网络的许多层里指定对神经元的期望响应，因为实际输出层在网络的时序行为展开时被重复很多次。

截断的通过时间的反向传播

为了使用通过时间的反向传播的实时形式，我们用误差平方和的瞬时值，即

$$\mathscr{E}(n) = \frac{1}{2}\sum_{j\in\mathscr{A}} e_j^2(n)$$

作为需要最小化的代价函数。如同标准反向传播学习的串行(随机)模式一样，我们使用代价函数$\mathscr{E}(n)$的负梯度去计算对于每个时刻 n 的网络的突触权值的适当调整量。当网络运行时，调整建立在连续的基础上。但是为了采用计算可行的方式，我们只在一个固定数目的时间步内储存相关的输入数据和网络状态的历史记录，该时间步数目称为*截断深度*(truncation depth)。此后截断深度用 h 表示。任何比 h 时间步早的信息是无关的，因此可以省略。如果不截断计算，由此容许回到开始时间，计算时间和储存要求当网络运行时会随时间线性增长，最终达到某点使得整个学习过程成为不可行的。

算法的第二种形式称为*截断的通过时间的反向传播*(truncated back-propagation-through-time, BPTT(h))算法(Williams and Peng, 1990)。神经元 j 的局部梯度定义为

$$\delta_j(l) = -\frac{\partial \mathscr{E}(l)}{\partial v_j(l)} \qquad 对于\ j \in \mathscr{A}\ 且\ n-h < l \leq n \tag{15.42}$$

由此导出公式

$$\delta_j(l) = \begin{cases} \varphi'(v_j(l))e_j(l) & 对于\ l = n \\ \varphi'(v_j(l)) \sum_{k \in \mathscr{A}} w_{kj}(l)\delta_k(l+1) & 对于\ n-h < l < n \end{cases} \tag{15.43}$$

一旦执行反向传播的计算到达时刻 $n-h+1$ 时，对神经元 j 的突触权值 w_{ji} 进行如下调整：

$$\Delta w_{ji}(n) = \eta \sum_{l=n-h+1}^{n} \delta_j(l)x_i(l-1) \tag{15.44}$$

754

其中 η 和 $x_i(l-1)$如前定义。注意式(15.43)中 $w_{kj}(l)$的使用需要保留权值的历史记录。只有当学习率参数 η 小到足以确保权值从一个时间步到下一时间步不会有很大改变的时候，在等式中使用 w_{kj} 才是合理的。

比较式(15.43)和(15.40)，可以看出与分回合的 BPTT 算法不同，误差信号只有在当前时间 n 才会进入计算。这就解释为什么不保存过去期望响应记录的原因。实际上，截断的通过时间的反向传播算法对前期时间步的处理，和随机反向传播算法(在第 4 章讨论)对待多层感知器中的隐藏神经元的计算是一样的。

一些实际考虑

在 BPTT 的实际应用中，截断并不是看起来那样是完全人为的。除非递归网络是不稳定的，对于导数$\partial\mathscr{E}(l)/\partial v_j(l)$应该收敛，这是因为时间上非常靠后的计算对应于更高的反馈能力(粗略地等于 sigmoid 斜率乘以权值)进行的。在任何情况下，截断深度 h 应该大到足以产生接近实际值的导数。这就要求值 h 有一个低的下界。例如，把动态驱动递归网络用于引擎慢速(idle-speed)控制时，$h = 30$ 是一个完成学习任务的相当保守的选择(Puskorius et al., 1996)。

另一实际问题需要讨论。本节讨论的通过时间的反向传播的展开过程提供一个利用相似层随时间前向处理的级联描绘它的有用工具，这样可以帮助我们深入理解过程是如何作用的。然而这个优点也是产生缺点的原因。在由很少神经元组成的相对简单的递归网络中过程运行良好。但是，当展开过程应用到那些实际中常遇到的更一般的结构时，基本公式，特别是式(15.43)，就变得笨拙。在这种情况下，更好的方法是用 Werbos (1990)描述的更一般的方法，此时每层的前向传播每一个表示引发一个相应的反向传播表示的集合。这个方法的优

点是对前向和递归(反馈)连接的相似处理。

为描述 BPTT(h)特殊形式的机理，令 F_{-x}^l 表示在节点 l 的网络输出对 x 的有序导数(ordered derivative)。为了导出反向传播方程，以相反的次序考虑前向传播方程。从每个方程根据下列原理推导一个或多个反向传播表达式：

755

$$\text{If} \quad a = \varphi(b, c), \text{then} \quad F_{-b}^l = \frac{\partial \varphi}{\partial b} F_{-a}^l \quad \text{and} \quad F_{-c}^l = \frac{\partial \varphi}{\partial c} F_{-a}^l \tag{15.45}$$

例 15.5 为了让有序导数的概念清晰，考虑下列两个方程的非线性系统：

$$x_1 = \log u + x_2^3$$

$$y = x_1^2 + 3x_2$$

变量 x_2 在两个方面影响输出 y：直接通过第二个方程，和间接通过第一个方程。y 对 x_2 的有序导数由包括 x_2 对 y 的直接和间接的作用效果的总因果影响所定义，可表示如下：

$$F_{-x_2} = \frac{\partial y}{\partial x_2} + \frac{\partial y}{\partial x_1}\frac{\partial x_1}{\partial x_2} = 3 + (2x_1)(3x_2^2) = 3 + 6x_1x_2^2$$

■

在编写程序时，对 BPTT(h)的有序导数，式(15.45)的右侧的每一个有序导数值被加到左侧的原来的值上。在这种方法中，适当的导数从网络中的一个给定的节点分配到了所有的以前向方式前馈该节点的网络其他节点和突触权值，并且对于每一连接中可能出现的延迟做出适当补偿。这里描述的表达式的简洁减少了对诸如时间展开或信号流图的可视化的需要。在 Feldkamp and Puskorius(1998)以及 Puskorius et al.(1996)中，利用这个过程产生了实现 BPTT 算法的伪代码。

15.8 实时递归学习

本节我们描述另一种称为*实时递归学习*(real-time recurrent learning, RTRL)[9] 的学习算法。算法的名称来自于下面的事实，完全连接网络的突触权值调整是实时的，也就是说，是在网络继续执行它的信号处理功能的时候(Williams and Zipser, 1989)。图 15-11 显示这样一个递归网络结构布局。它由 q 个神经元和 m 个外部输入组成。网络有两个不同的层：*并置的输入－反馈层*和*计算节点的处理层*。相应的，网络突触连接也是由前馈和反馈连接构成。

网络状态空间的描述由方程(15.10)和(15.11)定义。过程方程(15.10)重写成以下扩展形式：

756

$$\mathbf{x}(n+1) = \begin{bmatrix} \varphi(\mathbf{w}_1^T \boldsymbol{\xi}(n)) \\ \vdots \\ \varphi(\mathbf{w}_j^T \boldsymbol{\xi}(n)) \\ \vdots \\ \varphi(\mathbf{w}_q^T \boldsymbol{\xi}(n)) \end{bmatrix} \tag{15.46}$$

其中假设所有的神经元有相同的激活函数 $\varphi(\cdot)$。$(q+m+1)\times 1$ 向量 $\mathbf{w}_j$ 是递归网络的神经元 j 的突触权值向量，即

$$\mathbf{w}_j = \begin{bmatrix} \mathbf{w}_{a,j} \\ \mathbf{w}_{b,j} \end{bmatrix}, j = 1, 2, \cdots, q \tag{15.47}$$

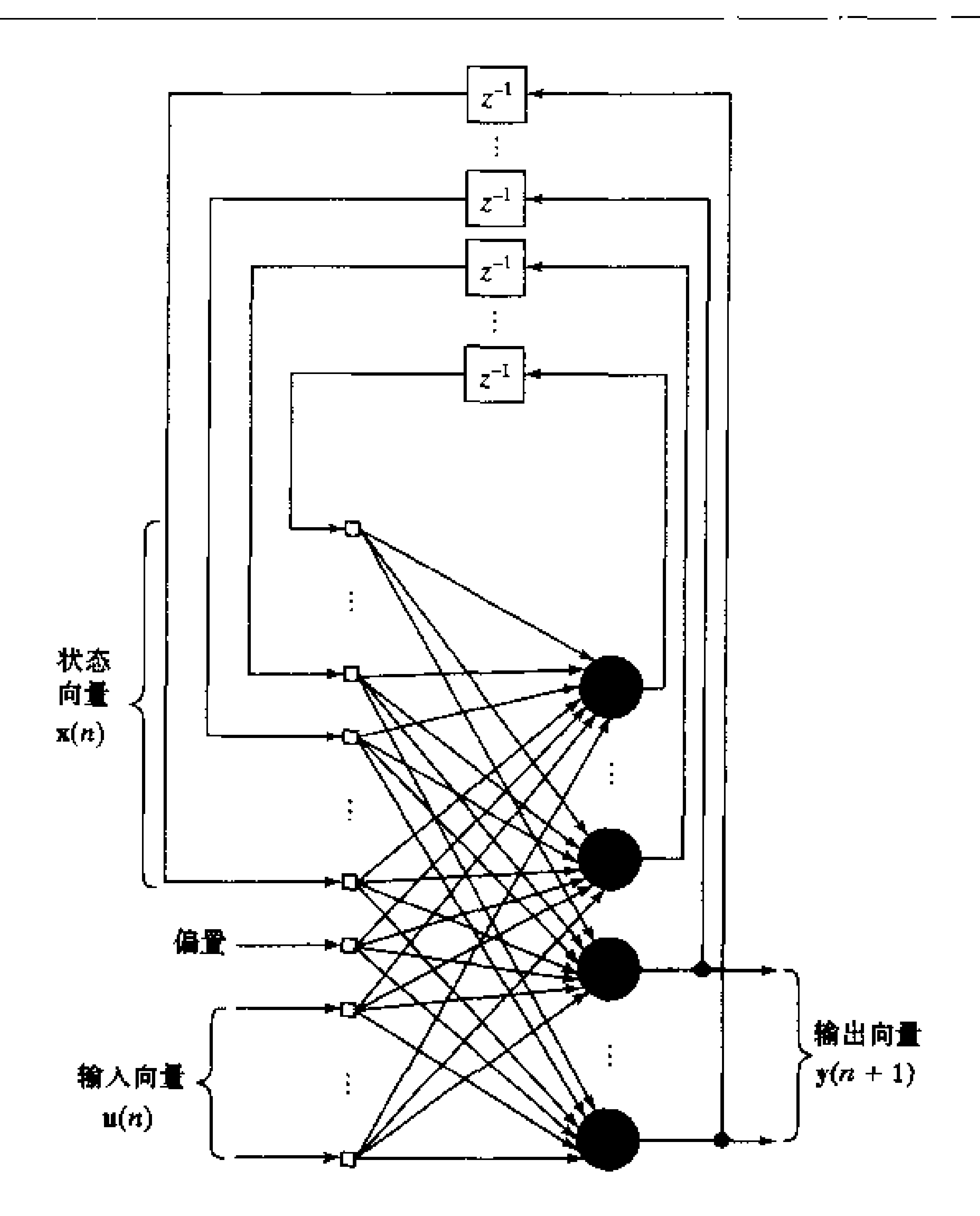


图 15-11 用于描述 RTRL 算法的完全连接递归网络

其中 $\mathbf{w}_{a,j}$和 $\mathbf{w}_{b,j}$分别是转置矩阵 $\mathbf{W}_a^T$ 和 $\mathbf{W}_b^T$ 的第j列。$(q+m+1)\times 1$ 向量 $\boldsymbol{\xi}(n)$定义为

$$\boldsymbol{\xi}(n)=\begin{bmatrix}\mathbf{x}(n)\\ \mathbf{u}(n)\end{bmatrix} \tag{15.48}$$

757

其中 $\mathbf{x}(n)$是 $q\times 1$ 状态向量，$\mathbf{u}(n)$是$(m+1)\times 1$ 输入向量。$\mathbf{u}(n)$的第一个元素是 $+1$，对应的 $\mathbf{w}_{b,j}$的第一个元素等于应用于神经元j的偏置 b_j。

为表达简单起见，引入新的矩阵 $\boldsymbol{\Lambda}_j(n)$，$\mathbf{U}_j(n)$和 $\boldsymbol{\Phi}(n)$，分别描述如下：

1.$\boldsymbol{\Lambda}_j(n)$是状态向量 $\mathbf{x}(n)$关于权值 $\mathbf{w}_j$ 的偏导数所构成的 $q\times(q+m+1)$矩阵：

$$\boldsymbol{\Lambda}_j(n)=\frac{\partial\mathbf{x}(n)}{\partial\mathbf{w}_j},j=1,2,\cdots,q \tag{15.49}$$

2.$\mathbf{U}_j(n)$是 $q\times(q+m+1)$矩阵，除了第j行等于向量 $\boldsymbol{\xi}(n)$外，其他行都为 0：

$$\mathbf{U}_j(n)=\begin{bmatrix}0\\ \boldsymbol{\xi}^T(n)\\ 0\end{bmatrix}\leftarrow 第\,j\,行,j=1,2,\cdots,q \tag{15.50}$$

3.$\boldsymbol{\Phi}(n)$是 $q\times q$ 的对角矩阵，它的第 k 个对角元素是激活函数对其自变量的偏导数，在 $\mathbf{w}_j^T\boldsymbol{\xi}(n)$处计算：

$$\mathbf{\Phi}(n) = \text{diag}(\varphi'(\mathbf{w}_1^T\boldsymbol{\xi}(n)),\cdots,\varphi'(\mathbf{w}_j^T\boldsymbol{\xi}(n)),\cdots,\varphi(\mathbf{w}_q^T\boldsymbol{\xi}(n))) \tag{15.51}$$

有了这些定义，就可以对式(15.46)关于 $\mathbf{w}_j$ 求导。用微积分的链式法则，得到下列递归公式：

$$\mathbf{\Lambda}_j(n+1) = \mathbf{\Phi}(n)[\mathbf{W}_a(n)\mathbf{\Lambda}_j(n) + \mathbf{U}_j(n)], \qquad j = 1,2,\cdots,q \tag{15.52}$$

这个递归公式描述实时递归学习过程的*非线性状态动力学*(即状态演化)。

为了完成描述这个学习过程，我们需要将矩阵 $\mathbf{\Lambda}_j(n)$和误差曲面对 $\mathbf{w}_j$ 的梯度相联系。为此，首先用度量方程(15.11)定义 $p\times 1$ 误差向量：

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n) = \mathbf{d}(n) - \mathbf{C}\mathbf{x}(n) \tag{15.53}$$

根据 $\mathbf{e}(n)$定义的平方误差瞬间和为

758

$$\mathscr{E}(n) = \frac{1}{2}\mathbf{e}^T(n)\mathbf{e}(n) \tag{15.54}$$

学习过程的目标是极小化由对所有时间 n 的$\mathscr{E}(n)$求和所得到的代价函数，即

$$\mathscr{E}_{\text{total}} = \sum_n \mathscr{E}(n)$$

为完成这个目标，使用最陡下降方法，这就需要*梯度矩阵*的知识，可写为

$$\nabla_{\mathbf{W}}\mathscr{E}_{\text{total}} = \frac{\partial \mathscr{E}_{\text{total}}}{\partial \mathbf{W}} = \sum_n \frac{\partial \mathscr{E}(n)}{\partial \mathbf{W}} = \sum_n \nabla_{\mathbf{W}}\mathscr{E}(n)$$

其中$\nabla_{\mathbf{W}}\mathscr{E}(n)$是$\mathscr{E}(n)$对权值矩阵 $\mathbf{W}=\{\mathbf{w}_k\}$的梯度。如果需要，可以继续使用这个方程并且得到递归网络的突触权值的更新方程，并且不用近似。但是，为了得到一个*实时的*训练递归网络使用的学习算法，必须使用一个梯度的瞬时估计值，即$\nabla_{\mathbf{W}}\mathscr{E}(n)$，这就导致对最陡下降方法的近似。

回到式(15.54)，以它作为最小化的代价函数，求它对权值向量 $\mathbf{w}_j$ 的微分，得到

$$\frac{\partial \mathscr{E}(n)}{\partial \mathbf{w}_j} = \left(\frac{\partial \mathbf{e}(n)}{\partial \mathbf{w}_j}\right)\mathbf{e}(n) = -\mathbf{C}\left(\frac{\partial \mathbf{x}(n)}{\partial \mathbf{w}_j}\right)\mathbf{e}(n) = -\mathbf{C}\mathbf{\Lambda}_j(n)\mathbf{e}(n), j = 1,2,\cdots,q \tag{15.55}$$

因此应用于神经元 j 的突触权值向量 $\mathbf{w}_j(n)$的调整由

$$\Delta\mathbf{w}_j(n) = -\eta\frac{\partial \mathscr{E}(n)}{\partial \mathbf{w}_j} = \eta\mathbf{C}\mathbf{\Lambda}_j(n)\mathbf{e}(n), j = 1,2,\cdots,q \tag{15.56}$$

决定，其中 η 是学习率参数，$\mathbf{\Lambda}_j(n)$由式(15.52)决定。

现在仅剩下确定开始学习过程的*初始条件*。为此令

759

$$\mathbf{\Lambda}_j(0) = \mathbf{0} \qquad \text{对所有 } j \tag{15.57}$$

这意味着递归网络的初始状态停留在一常态。

表15-1概括实时递归学习算法。这里所描述的算法公式可应用到任意的对其自变量可微的激活函数 $\varphi(\cdot)$。对于特殊情况，取双曲线切线方程形式的 sigmoid 非线性函数，我们有

$$x_j(n+1) = \varphi(v_j(n)) = \tanh(v_j(n))$$

且

$$\varphi'(v_j(n)) = \frac{\partial\varphi(v_j(n))}{\partial v_j(n)} = \text{sech}^2(v_j(n)) = 1 - x_j^2(n+1) \tag{15.58}$$

其中 $v_j(n)$神经元 j 的诱导局部域，$\mathbf{x}_j(n+1)$是它在 $n+1$ 时刻的状态。

表 15-1 实时递归学习算法小结

参数：

m = 输入空间维数

q = 状态空间维数

p = 输出空间维数

$\mathbf{w}_j$ = 神经元 j 的突触权值向量，$j=1, 2, \cdots, q$。

初始化：

1. 对算法的突触权值赋予从一个均匀分布中选出的较小值。

2. 置状态向量 $\mathbf{x}(0)$ 的初始值为 $\mathbf{x}(0)=\mathbf{0}$。

3. 对 $j=1, 2, \cdots, q$，置 $\mathbf{\Lambda}_j(0)=\mathbf{0}$。

计算：对 $n=0, 1, 2, \cdots$，计算

$$\mathbf{\Lambda}_j(n+1) = \mathbf{\Phi}(n)[\mathbf{W}_a(n)\mathbf{\Lambda}_j(n) + \mathbf{U}_j(n)]$$

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{C}\mathbf{x}(n)$$

$$\Delta\mathbf{w}_j(n) = \eta\mathbf{C}\mathbf{\Lambda}_j(n)\mathbf{e}(n)$$

$\mathbf{x}(n)$，$\mathbf{\Lambda}_j(n)$，$\mathbf{U}_j(n)$ 和 $\mathbf{\Phi}(n)$ 的定义分别由式(15.46)，(15.49)，(15.50)和(15.51)给出。

使用瞬时梯度 $\nabla_{\mathbf{w}}\mathscr{E}(n)$ 意味着实时递归学习算法偏离建立在真正梯度 $\nabla_{\mathbf{w}}\mathscr{E}_{\text{total}}$ 基础上的非实时算法。但是，该偏离和在第 4 章里用的训练多层感知器的反向传播算法很相似。虽然实时递归算法不保证和总的误差函数 $\mathscr{E}_{\text{total}}(\mathbf{W})$ 对权值矩阵 $\mathbf{W}$ 的负梯度精确一致，但实时和非实时的实际差别很小；在算法速率参数 η 减少时它们近似相等。与真正梯度偏离的行为所导致的潜在的最严重的结果，是观察的轨道(由绘制 $\mathscr{E}(n)$ 对权值矩阵 $\mathbf{W}(n)$ 的元素的图形获得)可能取决于算法产生的权值改变，这也可看作另一个反馈源并从而导致系统不稳定性。让参数 η 小到让权值变化的时间尺度远小于网络的运行的时间尺度，可以避免这个效果。 760

例 15.6 针对图 15-6 有两个输入和一个输出的完全递归网络，本例我们提出 RTRL 算法的公式。网络有三个神经元，由例 15.1 的矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{C}$ 构成。

由于 $m=2$，$q=3$，从式(15.48)可得

$$\boldsymbol{\xi}(n) = \begin{bmatrix} x_1(n) \\ x_2(n) \\ x_3(n) \\ 1 \\ u_1(n) \\ u_2(n) \end{bmatrix}$$

让 $\lambda_{j,kl}(n)$ 表示矩阵 $\mathbf{\Lambda}_j(n)$ 的第 kl 个元素。利用式(15.52)和(15.56)分别得到

$$\lambda_{j,kl}(n+1) = \varphi'(v_j(n))\left[\sum_{i=1}^{3} w_{ji}(n)\lambda_{j,kl}(n) + \delta_{kj}\xi_l(n)\right]$$

$$\Delta w_{kl}(n) = \eta(d_1(n)) - x_1(n))\lambda_{1,kl}(n)$$

其中 δ_{kj} 是 Kronecker delta，即 $k=j$ 时为 1，其他情况下为 0；$(j, k)=1, 2, 3$ 和 $l=1, 2, \cdots, 6$。图 15-12 表示一个决定权值调整 $\Delta w_{kl}(n)$ 演化的敏感度图。注意 $\mathbf{W}_a=\{w_{ji}\}$，$(j, i)=1, 2, 3$ 和 $\mathbf{W}_b=\{w_{jl}\}$，$j=1, 2, 3$，$l=4, 5, 6$。 761 ■

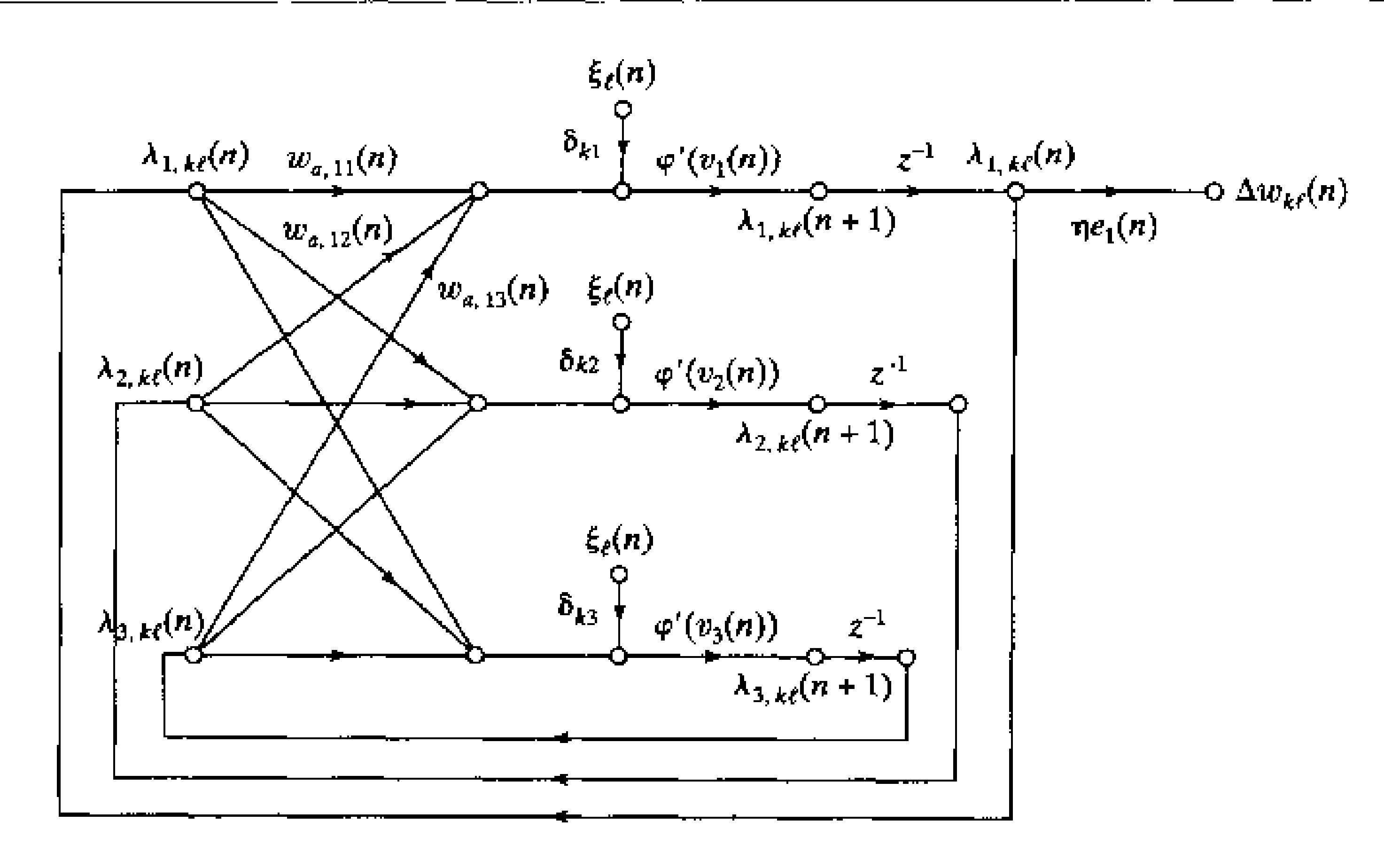


图 15-12 图 15-6 的全连接递归网络敏感度图

注意：标号为 $\xi_l(n)$ 的三个节点都看作单输入

教师强制

递归网络训练中经常用到的策略是*教师强制*(teacher forcing)(Williams and Zipser, 1989, 1995)；在自适应性滤波中，教师强制称为*方程－误差*(equation-error)*方法*(Mendel, 1995)。基本上教师强制涉及在网络的训练过程中每当期望响应可用时，在随后网络动态行为的计算中利用期望响应(即目标信号)替代实际神经元的输出。虽然教师强制是在 RTRL 算法下描述的，它的用法可以应用到另外的算法。但是，为了让它是可应用的，问题中的神经元必须将它的输出反馈回网络。

教师强制的良好效果包括(Williams and Zipser, 1995)：

- *教师强制可以使网络训练更快*。原因在于使用教师强制等于假设网络已经知道属于那些使用教师强制的神经元的任务的早期部分。
- *教师强制可以作为训练期的校正机制*。例如，网络的突触权值可能有正确的值，但是由于某种原因网络可能运行在状态空间的错误区域。显然在这种情况下，调整突触权值是错误的策略。

基于梯度的学习算法使用教师强制实际上是优化与不用教师强制不同的代价函数。教师强制算法和无强制算法产生不同的解，除非有关的误差信号为 0，这时勿需学习。

15.9 Kalman 滤波器

正如前面提及的一样，基于梯度下降的连续学习，例如实时递归学习算法，由于依赖梯度的瞬时估计，一般是很慢的。将递归网络的监督训练看做是*最优滤波问题*，可以克服这个严重的局限，它的方法是以回溯到学习过程的第一次迭代的方式递归利用包含在训练数据中的信息。这里描述的思想就是 *Kalman 滤波*的实质(Kalman, 1960)。Kalman 滤波器新颖的特点有：

- 理论是根据状态空间的概念提出的，可以有效利用包含在输入数据中的信息。
- 递归计算状态的估计；即每个更新的状态估计是依靠以前的估计和当前可用数据计算出的，因此只有以前的估计需要储存。

762

这一节我们给出 Kalman 滤波器理论[10]的简要回顾，便于下一节讨论解耦扩展 Kalman 滤波器。理论的发展常常开始于线性动态系统。为了扩展到非线性动态系统，将一个线性化的形式应用于系统；后一部分的讨论推迟到下一节。

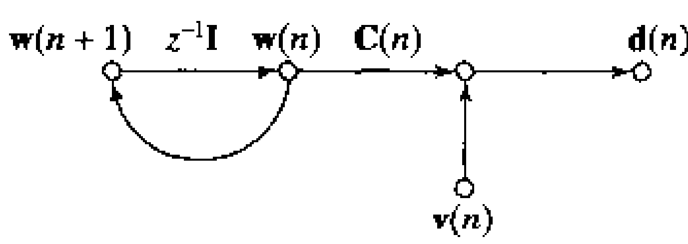


图 15-13 用于描述 Kalman 滤波器的线性离散时间动态系统信号流图

考虑图 15-13 的线性离散时间动态系统的信号流图。这里给出的系统的时间域描述和 15.3 节给出的状态空间形式相似。图 15-13 的数学表达式为如下方程：

$$\mathbf{w}(n+1)=\mathbf{w}(n) \tag{15.59}$$

$$\mathbf{d}(n)=\mathbf{C}(n)\mathbf{w}(n)+\mathbf{v}(n) \tag{15.60}$$

过程方程(process equation)(15.59)和度量方程(measurement equation)(15.60)的各个量如下：

- $\mathbf{w}(n)$是系统的状态向量
- $\mathbf{d}(n)$是观察向量
- $\mathbf{C}(n)$是度量矩阵
- $\mathbf{v}(n)$是度量噪声

在过程方程(15.59)中作了两个简化的假设。首先，过程方程是无噪声的。其次，系统在时刻 $n+1$ 和 n 的状态之间的转换矩阵等于单位矩阵。在图 15-13 我们使用了状态的一个新符号，其原因在下一节会十分明显。

Kalman 滤波问题可陈述如下：

利用由向量集$\{\mathbf{d}(i)\}_{i=1}^{n}$组成的所有观测数据，对于每一个 $n\geq 1$ 寻找状态 $\mathbf{w}(i)$的最小均差平方估计。

注意状态向量的信息是不可用的。如果 $i=n$，该问题称为滤波，如果 $i>n$，被称为预测，如果 $1\leq i\leq n$ 称为平滑。问题解的导出建立在下列假设的基础上(除了对系统线性性的假设)：

1. 度量噪声 $\mathbf{v}(n)$是均值为 0 的白噪声，其协方差矩阵定义为

$$E[\mathbf{v}(n)\mathbf{v}^T(k)]=\begin{cases}\mathbf{R}(n), & n=k\\ \mathbf{0}, & n\neq k\end{cases} \tag{15.61}$$

2. 对所有 $n\geq 0$，状态初始值 $\mathbf{w}(0)$与 $\mathbf{v}(n)$不相关。

763

为了得到 Kalman 滤波器的巧妙推导，我们将使用新息的概念(Kailath, 1968)。特别地，与观测向量 $\mathbf{d}(n)$有关的新息过程(innovations process)定义为

$$\boldsymbol{\alpha}(n)=\mathbf{d}(n)-\hat{\mathbf{d}}(n\mid n-1) \tag{15.62}$$

其中$\hat{\mathbf{d}}(n\mid n-1)$是$\mathbf{d}(n)$的最小均方误差估计，给出观测向量从时间 $n=1$ 开始并且扩展至时间 $n-1$ 的所有过去值。对于"最小均方误差估计"我们是指最小化对于 $\mathbf{d}(n)$测得的均方误差的特定估计。新息过程 $\boldsymbol{\alpha}(n)$可看做是包含在 $\mathbf{d}(n)$但不在 $\hat{\mathbf{d}}(n\mid n-1)$的预测部分的新信息的测量。新息过程 $\boldsymbol{\alpha}(n)$有如下的优点(Kailath, 1968)：

1. 与$\mathbf{d}(n)$有关的新息过程$\boldsymbol{\alpha}(n)$与过去的所有的观测值$\mathbf{d}(1),\mathbf{d}(2),\cdots,\mathbf{d}(n-1)$无关，即

$$E[\boldsymbol{\alpha}(n)\mathbf{d}^T(k)]=\mathbf{0} \quad 对于 1\leqslant k\leqslant n-1$$

2. 新息过程由一系列互相无关的随机向量构成，即

$$E[\boldsymbol{\alpha}(n)\boldsymbol{\alpha}^T(k)]=\mathbf{0} \quad 对于 1\leqslant k\leqslant n-1$$

3. 代表观测数据的随机向量序列和代表新息过程的随机向量序列一一对应，即

$$\{\mathbf{d}(1),\mathbf{d}(2),\cdots,\mathbf{d}(n)\}\rightleftharpoons\{\boldsymbol{\alpha}(1),\boldsymbol{\alpha}(2),\cdots,\boldsymbol{\alpha}(n)\} \tag{15.63}$$

现在不用损失任何信息我们就可以用不相关的新息序列代替相关的观测数据序列。给定新息集$\{\boldsymbol{\alpha}(k)\}_{k=1}^{n}$表示在时间$k$的状态估计。由此 Kalman 滤波器推导变得简单了。在此基础上进行分析，我们可以导出标准 Kalman 滤波器，如表 15-2 中的小结。

表 15-2 Kalman 滤波器小结

对 $n=1,2,3,\cdots$，计算

$$\boldsymbol{\Gamma}(n)=[\mathbf{C}(n)\mathbf{K}(n,n-1)\mathbf{C}^T(n)+\mathbf{R}(n)]^{-1}$$
$$\mathbf{G}(n)=\mathbf{K}(n,n-1)\mathbf{C}^T(n)\boldsymbol{\Gamma}(n)$$
$$\boldsymbol{\alpha}(n)=\mathbf{y}(n)-\mathbf{C}(n)\hat{\mathbf{w}}(n\mid n-1)$$
$$\hat{\mathbf{w}}(n+1\mid n)=\hat{\mathbf{w}}(n\mid n-1)+\mathbf{G}(n)\boldsymbol{\alpha}(n)$$
$$\mathbf{K}(n+1,n)=\mathbf{K}(n,n-1)-\mathbf{G}(n)\mathbf{C}(n)\mathbf{K}(n,n-1)$$

764

这里有三个新的量需要定义：

- $\mathbf{K}(n,\ n-1)$是*误差协方差矩阵*，定义为

$$\mathbf{K}(n,n-1)=E[\boldsymbol{\varepsilon}(n,n-1)\boldsymbol{\varepsilon}^T(n,n-1)] \tag{15.64}$$

  其中状态误差$\boldsymbol{\varepsilon}(n,\ n-1)$定义如下

$$\boldsymbol{\varepsilon}(n,n-1)=\mathbf{w}(n)-\hat{\mathbf{w}}(n\mid n-1) \tag{15.65}$$

  其中$\mathbf{w}(n)$是实际状态，$\hat{\mathbf{w}}(n\mid n-1)$是建立在直到时间$n-1$为止的过去观测数据基础上的单步预测值。

- $\boldsymbol{\Gamma}(n)$是关于滤波估计误差$\mathbf{e}(n)$和新息$\boldsymbol{\alpha}(n)$关联的*转换因子*(conversion factor)，即

$$\mathbf{e}(n)=\mathbf{R}(n)\boldsymbol{\Gamma}(n)\boldsymbol{\alpha}(n) \tag{15.66}$$

  其中

$$\mathbf{e}(n)=\mathbf{d}(n)-\hat{\mathbf{d}}(n\mid n) \tag{15.67}$$

  $\hat{\mathbf{d}}(n\mid n)$是在直到时间$n$为止的观测数据下的观测向量$\mathbf{d}(n)$的估计。

- $\mathbf{G}(n)$是 Kalman *增益*(gain)，用于决定更新状态估计的校正量。

表 15-2 小结的 Kalman 滤波器类型被设计用于传播误差的协方差矩阵$\mathbf{K}(n,n-1)$。因此这个算法称为*协方差 Kalman 滤波算法*(covariance Kalman filtering algorithm)。

平方根 Kalman 滤波器

协方差 Kalman 滤波器会有严重的数值困难。特别当更新矩阵$\mathbf{K}(n+1,n)$是由 Riccati 方程决定时，它在表 15-2 的最后一行定义。*Riccati 方程*的右边是两个矩阵量的差。除非在算法的每一次迭代中使用的数值精确度都足够高，否则从这个计算所得到的更新矩阵$\mathbf{K}(n+1,n)$可能不为非负定的。很明显这样的解是不可接受的，因为$\mathbf{K}(n+1,n)$代表协方差矩阵，由定义它是非负定的。由于使用有限字长算术而产生的数值不准确性，进而导致 Kalman 滤波器的非稳定行为称为*发散现象*(divergence phenomenon)。

这个问题可以通过传播误差协方差矩阵的平方根 $\mathbf{K}^{1/2}(n,n-1)$ 而不是 $\mathbf{K}(n,n-1)$ 自身来解决。具体地，使用 Cholesky 因式分解，我们将 $\mathbf{K}(n,n-1)$ 表示为(Golub and Van Loan, 1996)：

$$\mathbf{K}(n,n-1)=\mathbf{K}^{1/2}(n,n-1)\mathbf{K}^{T/2}(n,n-1) \tag{15.68}$$

这里 $\mathbf{K}^{1/2}(n,n-1)$ 是一个下三角矩阵，$\mathbf{K}^{T/2}(n,n-1)$ 是它的转置。在线性代数中，Cholesky 因子 $\mathbf{K}^{1/2}(n,n-1)$ 通常指的是 $\mathbf{K}(n,n-1)$ 的平方根。因此建立在 Cholesky 因式分解基础上的 Kalman 滤波器被称为平方根 Kalman 滤波器[11]。重要的一点是，矩阵的乘积 $\mathbf{K}^{1/2}(n,n-1)\mathbf{K}^{T/2}(n,n-1)$ 为不确定的可能性大大减少，因为任何方阵和它转置矩阵的乘积总是正定的。

15.10 解耦扩展的 Kalman 滤波器

我们对 Kalman 滤波器的主要兴趣在于利用它的独有特性来执行递归网络的监督训练[12]。由于递归网络结构的复杂性(例如递归多层感知器)，问题关键在于如何在不损害 Kalman 滤波器理论应用的同时又让该方法计算上可行。找到的答案是使用一个扩展 Kalman 滤波器的解耦形式，其计算的复杂性适应于可利用的计算资源和和特定的应用(Puskorius and Feldkamp, 1991)。 765

考虑建立在具有 W 个突触权值和 p 个输出节点的静态多层感知器基础上的递归网络。令向量 $\mathbf{w}(n)$ 表示在时间 n 时整个网络的突触权值。根据自适应滤波器的思想，网络的状态空间方程可以建模如下(Singhal and Wu, 1989; Haykin, 1996)：

$$\mathbf{w}(n+1)=\mathbf{w}(n) \tag{15.69}$$

$$\mathbf{d}_o(n)=\mathbf{c}(\mathbf{w}(n),\mathbf{u}(n),\mathbf{v}(n))+\mathbf{v}(n) \tag{15.70}$$

这里权值向量 $\mathbf{w}(n)$ 起到状态的作用。属于向量值函数 $\mathbf{c}(\cdot,\cdot,\cdot)$ 的第二个向量参数 $\mathbf{u}(n)$ 和第三个向量参数 $\mathbf{v}(n)$ 分别表示输入向量和回归节点激活的向量。实际上式(15.69)指出模型停留于最佳状态，转换矩阵在时间 n 将 $\mathbf{w}(n)$ 转换为在时间 $n+1$ 的 $\mathbf{w}(n+1)$，它是单位矩阵。最佳条件是指递归网络误差曲面的局部或全局最小。模型非线性的惟一来源是度量方程(15.70)。向量 $\mathbf{d}_o$ 表示模型的期望响应。由于式(15.70)表示模型的输入－输出方程，可知 $\mathbf{c}(\cdot,\cdot,\cdot)$ 表示多层感知器的输入层到输出层的整个非线性性；式(15.70)的噪声度量向量 $\mathbf{v}(n)$ 假设是一个 0 均值和对角协方差矩阵 $\mathbf{R}(n)$ 的多元白噪声过程。

在应用扩展的 Kalman 滤波器到递归网络时，必须注意"状态"是在两种不同的环境下使用的术语：

- 系统演化通过自适应性滤波，这显示在训练中对递归网络权值的改变；向量 $\mathbf{w}(n)$ 表示这第一种状态概念。
- 递归网络自身的运行，例如函数 $\mathbf{c}$ 所依赖的回归节点激活；向量 $\mathbf{v}(n)$ 表示这第二种状态概念。

通过比较式(15.69)和(15.70)描述的模型与式(15.59)和(15.60)的线性动态模型，可以看到这两个模型的惟一差别在于度量方程的非线性的形式。为了应用 Kalman 滤波器理论到刚描述的状态空间模型，我们必须首先线性化式(15.70)，并改写为

$$\mathbf{d}(n)=\mathbf{C}(n)\mathbf{w}(n)+\mathbf{v}(n) \tag{15.71}$$

766

的形式，其中 $\mathbf{C}(n)$ 是线性模型 $p\times W$ 的度量矩阵，用 $\mathbf{d}(n)$ 区别于式(15.70)的 $\mathbf{d}_o(n)$。线性化包括整个网络的 p 个输出对模型 W 个权值的偏微分，表示为

$$\mathbf{C}(n) = \begin{bmatrix} \frac{\partial c_1}{\partial w_1} & \frac{\partial c_1}{\partial w_2} & \cdots & \frac{\partial c_1}{\partial w_W} \\ \frac{\partial c_2}{\partial w_1} & \frac{\partial c_2}{\partial w_2} & \cdots & \frac{\partial c_2}{\partial w_W} \\ \vdots & \vdots & & \vdots \\ \frac{\partial c_p}{\partial w_1} & \frac{\partial c_p}{\partial w_2} & \cdots & \frac{\partial c_p}{\partial w_W} \end{bmatrix} \tag{15.72}$$

其中 c_i，$i=1, 2, \cdots, p$ 表示非线性函数 $\mathbf{c}(\mathbf{w}(n), \mathbf{u}(n), \mathbf{v}(n))$的第 i 个元素。式(15.72)的偏微分在 $\mathbf{w}(n)=\hat{\mathbf{w}}(n)$处计值，其中 $\hat{\mathbf{w}}(n)$是在时刻 n 权值向量 $\mathbf{w}(n)$的估值，它由扩展的 Kalman 滤波器在给出直到时刻 $n-1$ 的观察数据基础上计算出来(Haykin, 1996)。在实现时，这些偏微分是由通过时间的反向传播算法或实时递归学习算法计算出来的。实际上，扩展的 Kalman 滤波器算法建立在 15.7 节或 15.8 节中提到的这两种算法中的一个或另一个的基础上。这意味着 $\mathbf{c}$ 必须是一个关于刚才提到的递归节点激活的函数。事实上，对于单层递归网络，矩阵 $\mathbf{C}(n)$能够由矩阵 $\mathbf{\Lambda}_j(n)$的元素组成，就像式(15.52)中的 RTRL 算法所计算的一样。因此，度量矩阵 $\mathbf{C}(n)$是网络输出对网络自由参数的动态导数矩阵。正像在时间步$(n+1)$时网络递归节点的激活是一个对前面的时间步 n 得到的相应值的函数一样，按照相似的方法，我们发现在时间步$(n+1)$时，递归节点激活对网络自由参数的导数就像在 RTRL 方程所表示的那样，为前面的时间步 n 得到的相应值的函数。

假设网络的突触权值被分为 g 组，例如，第 i 组有 k_i 个神经元。在式(15.72)定义的 $p\times W$ 度量矩阵 $\mathbf{C}$ 是网络输出对所有网络权值的导数矩阵。矩阵 $\mathbf{C}(n)$对于输入向量 $\mathbf{u}(n)$的依赖关系由式(15.72)所隐含定义。这样定义的矩阵 $\mathbf{C}(n)$包括对于扩展的 Kalman 滤波器的任何解耦形式所必需的导数。例如，如果使用*全局扩展* Kalman *滤波器*(global extended Kalman filter, GEKF)(即我们没有解耦)，$g=1$，并且整个矩阵 $\mathbf{C}(n)$由式(15.72)所定义。在另一方面，如果使用*解耦扩展* Kalman *滤波器*(decoupled extended Kalman filter, DEKF)，那么“全局”度量矩阵 $\mathbf{C}(n)$必须调整使得网络中一个给定的神经元的权值被分在一个组，在 $\mathbf{C}(n)$内部作为一个单独块，其中每一个块被标记为 $i=1, 2, \cdots, g$。对于后者，矩阵 $\mathbf{C}(n)$仅仅是单个 $\mathbf{C}_i$ 的**并置**，如下面所示：

767

$$\mathbf{C}(n) = [\mathbf{C}_1(n), \mathbf{C}_2(n), \cdots, \mathbf{C}_g(n)]$$

不管解耦程度如何，整个矩阵 $\mathbf{C}(n)$必须如式(15.72)所定义的那样计算。

现在开始应用表 15-2 的 Kalman 滤波器算法。特别地，对于式(15.69)和(15.71)的线性化动态模型，我们有(Puskorius and Feldkamp, 1991)：

$$\mathbf{\Gamma}(n) = \left[\sum_{i=1}^{g}\mathbf{C}_i(n)\mathbf{K}_i(n,n-1)\mathbf{C}_i^T(n) + \mathbf{R}(n)\right]^{-1} \tag{15.73}$$

$$\mathbf{G}_i(n) = \mathbf{K}_i(n,n-1)\mathbf{C}_i^T(n)\mathbf{\Gamma}(n) \tag{15.74}$$

$$\boldsymbol{\alpha}(n) = \mathbf{d}(n) - \hat{\mathbf{d}}(n \mid n-1) \tag{15.75}$$

$$\hat{\mathbf{w}}_i(n+1 \mid n) = \hat{\mathbf{w}}_i(n \mid n-1) + \mathbf{G}_i(n)\boldsymbol{\alpha}(n) \tag{15.76}$$

$$\mathbf{K}_i(n+1,n) = \mathbf{K}_i(n,n-1) - \mathbf{G}_i(n)\mathbf{C}_i(n)\mathbf{K}_i(n,n-1) \tag{15.77}$$

其中 $i=1, 2, \cdots, g$。式(15.73)至(15.77)的参数向量和信号向量描述如下：

$\mathbf{\Gamma}(n) = p\times p$ 矩阵，表示整个网络的全局转换因子

$\mathbf{G}_i(n) = W_i \times p$ 矩阵，表示第 i 组神经元的 Kalman 增益

$\boldsymbol{\alpha}(n) = p \times 1$ 向量，表示线性化系统的期望响应 $\mathbf{d}(n)$ 和它的估计 $\hat{\mathbf{d}}(n|n-1)$ 的差值

估计 $\hat{\mathbf{d}}(n|n-1)$ 由网络停留在状态 $\{\hat{\mathbf{w}}_i(n|n-1)\}$ 时网络的实际输出 $\mathbf{y}(n)$ 表示，实际输出 $\mathbf{y}(n)$ 为网络对输入 $\mathbf{u}(n)$ 产生的响应

$\hat{\mathbf{w}}_i(n|n-1) = W \times 1$ 向量，表示在给定直到时间 $n-1$ 为止的观察数据情况下，对于第 i 组的权值矩阵 $\mathbf{w}_i(n)$ 的估计

$\mathbf{K}_i(n, n-1) = k_i \times k_i$ 矩阵，表示第 i 组神经元的误差协方差矩阵

包括在(15.73)的全局转换因子 $\boldsymbol{\Gamma}(n)$ 定义中的求和说明扩展的 Kalman 滤波器的解耦本质。

很重要的一点，就是理解在 DEKF 算法里解耦实际决定全局误差协方差矩阵 $\mathbf{K}(n, n-1)$ 中哪些特定元素需要保持和更新。实际上，所有计算的节省是由于忽略与全局误差协方差矩阵 $\mathbf{K}(n, n-1)$ 的那些非对角块有关的保持和更新。

由式(15.73)至(15.77)编码的 DEKF 算法最小化代价函数

$$\mathscr{E}(n) = \frac{1}{2} \sum_{j=1}^{n} \| \mathbf{e}(j) \|^2 \qquad (15.78)$$

768

这里 $\mathbf{e}(j)$ 是误差向量，定义为

$$\mathbf{e}(j) = \mathbf{d}(j) - \mathbf{y}(j), \qquad j = 1, 2, \cdots, n$$

$\mathbf{y}(j)$ 是网络使用直到时间 j（包括时间 j）的所有可用信息的实际输出。注意，一般情况下，$\mathbf{e}(j) \neq \boldsymbol{\alpha}(j)$。

人工过程噪声

式(15.69)至(15.70)的非线性动态系统是*非强制的*，即过程方程(15.69)没有外部输入。这个缺陷可能导致严重的数值困难，因此在有限精度环境运行时产生 Kalman 滤波器发散。如 15.9 节解释的，发散现象可以用平方根滤波解决。

另一规避发散现象的方法是使用启发式的机制，涉及对过程方程人为添加过程噪声，表示为

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \boldsymbol{\omega}_i(n), \qquad i = 1, 2, \cdots, g \qquad (15.79)$$

其中 $\boldsymbol{\omega}_i(n)$ 即过程噪声。假设 $\boldsymbol{\omega}_i(n)$ 是一零均值和对角协方差矩阵为 $\mathbf{Q}_i(n)$ 的多变量白噪声。人为添加过程噪声 $\boldsymbol{\omega}_i(n)$ 实际上是与度量噪声 $\mathbf{v}(n)$ 和网络初始状态独立的。添加 $\boldsymbol{\omega}_i(n)$ 到式(15.79)所得到的效果是修改用于误差协方差矩阵更新的 Riccati 方程如下（Haykin, 1996）：

$$\mathbf{K}_i(n+1, n) = \mathbf{K}_i(n, n-1) - \mathbf{G}_i(n)\mathbf{C}_i(n)\mathbf{K}_i(n, n-1) + \mathbf{Q}_i(n) \qquad (15.80)$$

假设 $\mathbf{Q}_i(n)$ 对于所有的 i 都足够大，于是 $\mathbf{K}_i(n+1, n)$ 对于所有的 n 都是非负定的。

除了克服数值上的困难，人为添加过程噪声 $\boldsymbol{\omega}_i(n)$ 还有下列有益效果：在训练过程中，算法过程有较小可能性陷入局部最小。这就导致在收敛速度和解的质量方面使训练性能显著提高。

DEKF 算法小结

表 15-3 表示在式(15.73)至(15.76)及(15.80)基础上的 DEKF 算法小结。这个表也包括

算法的初始化细节。

现在可以对扩展的 Kalman 滤波器作最终评价如下。表 15-3 小结的 DEKF 算法指的是所有可能的信息保持学习过程(information - preserving learning procedure)的整个算法族，包括 GEKF。作为一般的法则，在解的质量方面我们期望 DEKF 产生的性能能接近 GEKF 但不希望超过它的性能。另一方面，DEKF 计算上要求比 GEKF 要少。虽然 DEKF 有计算上的优点，现在计算机速度和内存的增加使得 GEKF 对于特定的问题的计算成为可能，特别是在递归网络的离线训练时。

769

表 15-3 DEKF 算法小结

初始化：

1. 对递归网络的突触权值赋予从一个均匀分布中选出的较小值。
2. 置协方差矩阵 $\mathbf{Q}(n)$(表示人为插入的过程噪声 $\boldsymbol{\omega}(n)$)等于 10^{-6}到 10^{-2}。
3. $\mathbf{K}(1,0)=\delta^{-1}\mathbf{I}$，$\delta$ = 小的正常数。

计算：

对 $n=1, 2, \cdots$，计算

$$\boldsymbol{\Gamma}(n)=\left[\sum_{i=1}^{g}\mathbf{C}_i(n)\mathbf{K}_i(n,n-1)\mathbf{C}_i^T(n)+\mathbf{R}(n)\right]^{-1}$$

$$\mathbf{G}_i(n)=\mathbf{K}_i(n,n-1)\mathbf{C}_i^T(n)\boldsymbol{\Gamma}(n)$$

$$\boldsymbol{\alpha}(n)=\mathbf{d}(n)-\hat{\mathbf{d}}(n\mid n-1)$$

$$\hat{\mathbf{w}}_i(n+1\mid n)=\hat{\mathbf{w}}_i(n\mid n-1)+\mathbf{G}_i(n)\boldsymbol{\alpha}(n)$$

$$\mathbf{K}_i(n+1,n)=\mathbf{K}_i(n,n-1)-\mathbf{G}_i(n)\mathbf{C}_i(n)\mathbf{K}_i(n,n-1)+\mathbf{Q}_i(n)$$

其中第三行 $\hat{\mathbf{d}}(n\mid n-1)$为网络对输入向量 $\mathbf{u}(n)$产生的实际输出向量 $\mathbf{y}(n)$。

注：对 $g=1$(即无解耦)，DEKF 算法变为全局扩展的 Kalman 滤波(GEKF)算法

计算复杂性

表 15-4 提出本章所讨论的三种学习算法计算复杂性的比较：通过时间的反向传播，实时递归学习，解耦扩展 Kalman 滤波器。它们计算复杂性依次增加。

表 15-4 用于递归网络的学习算法的计算复杂性比较

S = 状态数
W = 突触权值数
L = 训练序列长度

1. 通过时间的反向传播(BPTT)
   - 时间需求，存储空间需求：$O(WL+SL)$，$O(WL+SL)$
2. 实时递归学习(RTRL)算法
   - 时间需求，存储空间需求：$O(WS^2L)$，$O(WS)$
3. 解耦扩展 Kalman 滤波(DEKF)算法：
   - 在最小值处，DEKF 利用 RTRL 或 BPTT 计算导数(在时间和空间)花费同样的代价；对 BPTT，时间和空间要求为网络输出数 p 乘以标准 BPTT 计算单一标量误差项的导数所花的代价。
   - 另外，DEKF 要求的时间复杂性为 $O(p^2W+p\sum_{i=1}^{g}k_i^2)$以及存储空间为 $O(\sum_{i=1}^{g}k_i^2)$，其中 g 为组数而 k_i 为第 i 组神经元数。当只有一个权值组时，如同在 GEKF 中一样，时间和空间存储需求分别变为 $O(pW^2)$和 $O(W^2)$。

15.11 计算机实验

这个实验再次讨论 13.5 节研究的非线性时间序列的模拟。时间序列由频率调制信号定义：

$$x(n)=\sin(n+\sin(n^2)) \qquad n=0,1,2,\cdots$$

我们将研究用于模拟的两种不同结构：

- 递归多层感知器(recurrent multilayer perceptron, RMLP)有 1 个输入节点，10 个递归神经元的第一隐藏层，10 个神经元的第二隐藏层和 1 个线性输出神经元。 770
- 集中时滞前馈网络(focused time lagged feedforward network, TLFN)，包括 20 个抽头延迟时间记忆，和由 10 个隐含神经元和 1 个线性输出神经元组成的多层感知器。

RMLP 比集中 TLFN 有稍多的突触权值，但是只有它一半的记忆(10 个递归节点和 20 个抽头)。

利用 DEKF 算法对 RMLP 进行训练。利用扩展的 Kalman 滤波器的两种形式对 TLFN 进行训练：(1)GEKF 算法(即全局形式)，(2)DEKF 算法(即解耦形式)。两个算法的细节如下：

- *GEKF*

  δ = 用作初始化误差协方差矩阵 $\mathbf{K}(n, n-1)$ 的参数

  $=0.01$

  $\mathbf{R}(n)$ = 度量噪声 $\mathbf{v}(n)$ 的协方差矩阵：开始训练时 $\mathbf{R}(0)=100$，在训练结束时退火至 $\mathbf{R}(n)=3$

  $\mathbf{Q}(n)$ = 人工过程噪声 $\boldsymbol{\omega}(n)$ 的协方差矩阵：开始时 $\mathbf{Q}(0)=10^{-2}$，在训练结束时退火至 $\mathbf{Q}(n)=10^{-6}$

  $\mathbf{R}(n)$ 和 $\mathbf{Q}(n)$ 的退火在训练过程中起到加快学习速度的作用。

- *DEKF*

  g = 组数

  $$=\begin{cases}21 & \text{用于 RMLP}\\ 11 & \text{用于集中 TLFN}\end{cases}$$

  其他参数和 GEKF 的一样

训练是在 4000 个样本序列上进行的。对于 RMLP，使用了长度为 100 的子集，在整个训练过程中，处理 30 000 个子集。具有 4000 个样本的训练集中的每个数据点处理了大约 750 次。对于集中 TLFN，在训练集中的每个数据点也处理了约 750 次。在两种情况下，测试都对 300 个数据点进行。

图 15-14 表示利用 DEKF 算法训练的 RMLP 计算出的单步预测波形 $\hat{y}(n)$。这个图也包括实际的波形 $y(n)$。两波形很难区分。图 15-15a 显示由 RMLP 产生的预测误差 771

$$e(n)=y(n)-\hat{y}(n)$$

相应的由算法 GEKF 和 DEKF 训练的集中 TLFN 产生的预测误差分别显示在图 15-15b 和 15-15c。通过比较图 15-15 的结果及 13.5 节的模拟结果，可以得到如下观察结果：

1. 均方误差意义上最精确的模拟是由 DEKF 算法训练的 RMLP 得到；对 5980 个样本计算的预测误差的方差是 1.1839×10^{-4}。

2. 对于集中 TLFN，均方误差意义上的最精确的模拟是通过 GEKF 训练得到的。对于 GEKF 训练，预测误差的方差是 1.3351×10^{-4}，而对于 GEKF 训练，预测误差的方差是 1.5871×10^{-4}。两个都是用 5980 个样本计算的。

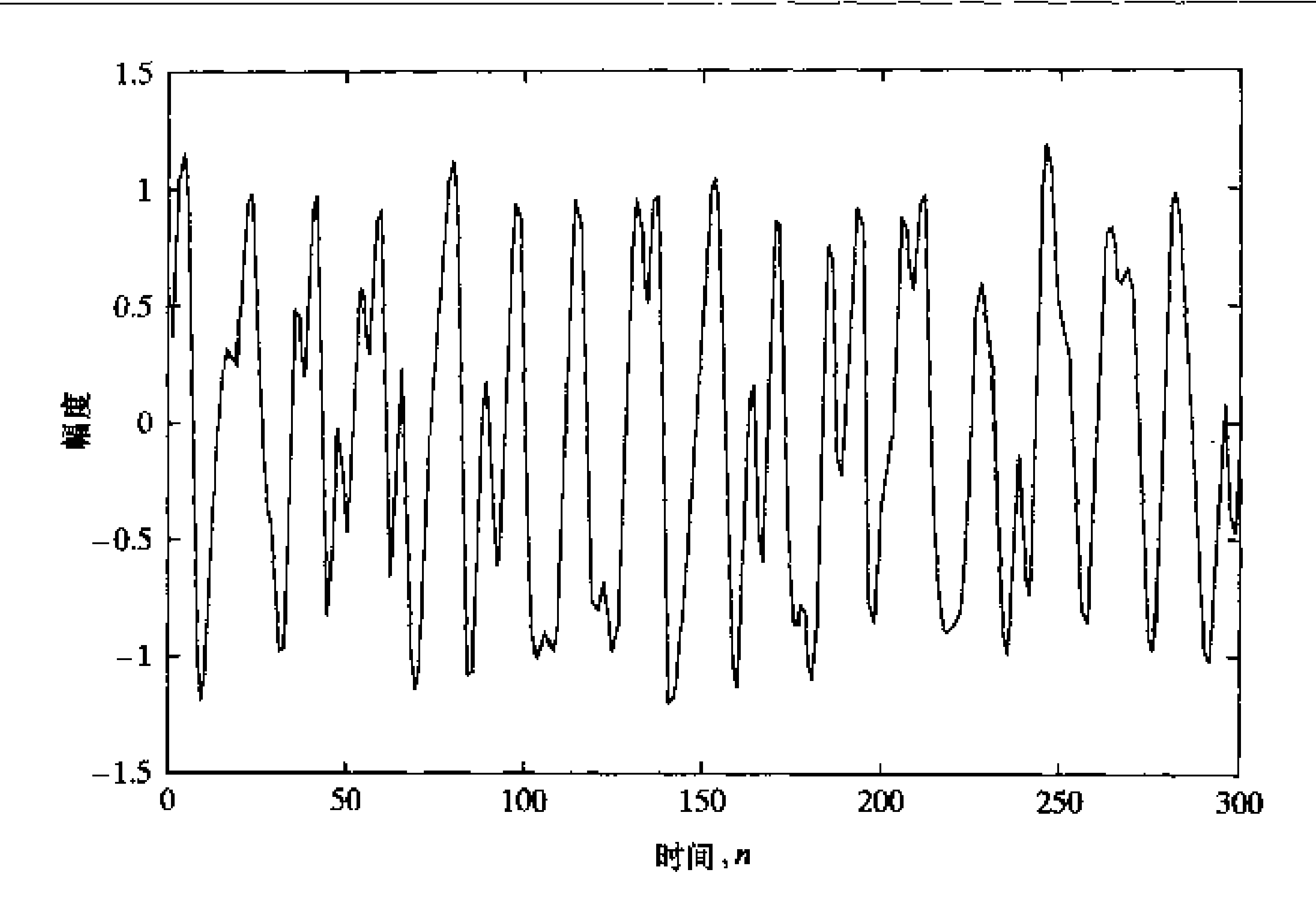
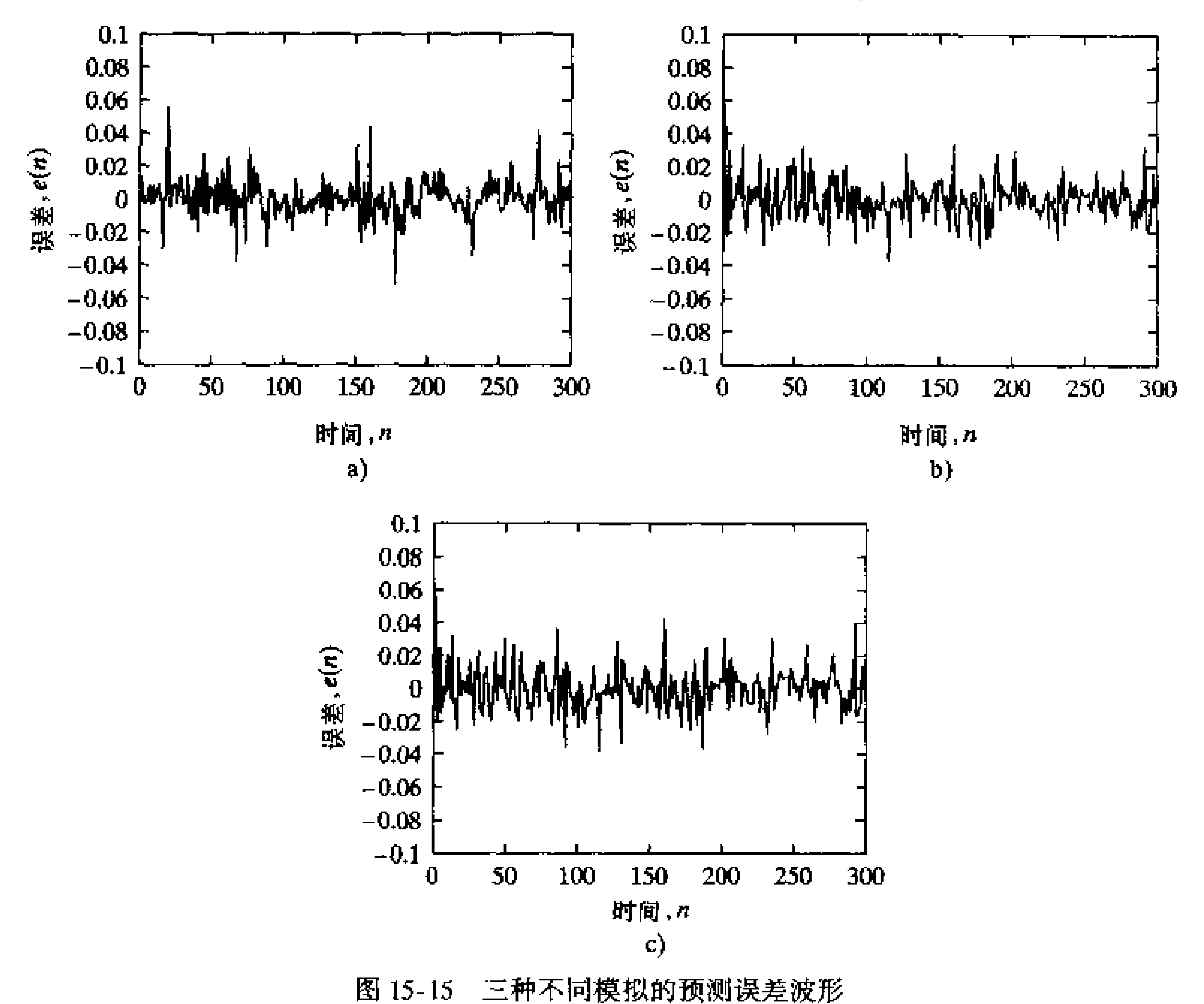


图 15-14 计算机建模试验的实际波形(实线)和预测波形(虚线)叠加图，利用 DEKF 算法训练的 RMLP 所计算预测波形



772

图 15-15 三种不同模拟的预测误差波形

a)由 DEKF 训练的 RMLP，误差方差 1.1839×10^{-4} b)由 GEKF 训练的 TLFN，误差方差 = 1.3351×10^{-4}
c)由 DEKF 训练的聚焦 TLFN，误差方差 = 1.5871×10^{-4}

3. 对于利用标准反向传播算法训练的集中 TLFN，第 13.5 节报告的预测误差的方差是 1.2×10^{-3}。这比由 CEKF 算法和 DEKF 算法得到的结果要差一个数量级。

相对反向传播而言扩展 Kalman 滤波器的优异的学习性能归因于它的信息保持性。

15.12 递归网络的消失梯度

递归网络的实际应用需要引起注意的一个问题是*消失梯度*(vanishing gradient)，它和依靠很久以前的输入数据用来训练网络使之在当前时刻产生一个期望响应有关(Hochreiter, 1991; Bengio et al., 1994)。关键是由于组合的非线性性，一个时间上隔得远的输入的一个微小变化对网络的训练几乎不会产生影响。即使时间上隔得远的输入的大的变化产生影响，但影响不能被梯度检测到，这时问题同样可能出现。消失梯度问题在一些特定情况下使得基于梯度的训练算法中长期依赖的学习即使不是完全不可能也是变得很困难。 [773]

在 Bengio et al.,(1994)中，对许多实际应用曾经讨论过，需要递归网络能够存储任意时间长度的状态信息，而在有噪声的情况下是否有必要这样做。在递归网络状态变量中长期存储的有限位的信息称为*信息锁存*(information latching)。信息锁存必须*很鲁棒*，不能被与当前学习任务无关的事件删除。用特殊术语，我们可以陈述如下(Bengio et al., 1994)：

如果网络状态包含在一个双曲吸引子的压缩吸引集中，则递归网络的鲁棒性信息锁存就可以实现。

双曲吸引子的概念在 14 章讨论。一个双曲吸引子的压缩集是在吸引盆的一个点集合，在这些点处 Jacobi 矩阵的所有特征值的绝对值小于 1。这就意味着如果递归网络的状态 $\mathbf{x}(n)$ 在一个双曲吸引盆，而不在压缩吸引集中，那么在 $\mathbf{x}(n)$ 周围的一个不确定球(ball of uncertainty)的大小会随时间而指数增长，如图 15-16a 所示。所以，对于递归网络输入的小扰动(噪声)能够将轨道推向另一个(可能是错的)吸引盆。但是如果状态 $\mathbf{x}(n)$ 继续保持在双曲吸引子的压缩吸引集中，这时在输入 $\mathbf{x}(n)$ 能够找到一个有界范围使得 $\mathbf{x}(n)$ 停留在吸引子的一定距离之内，如图 15-16 所示。

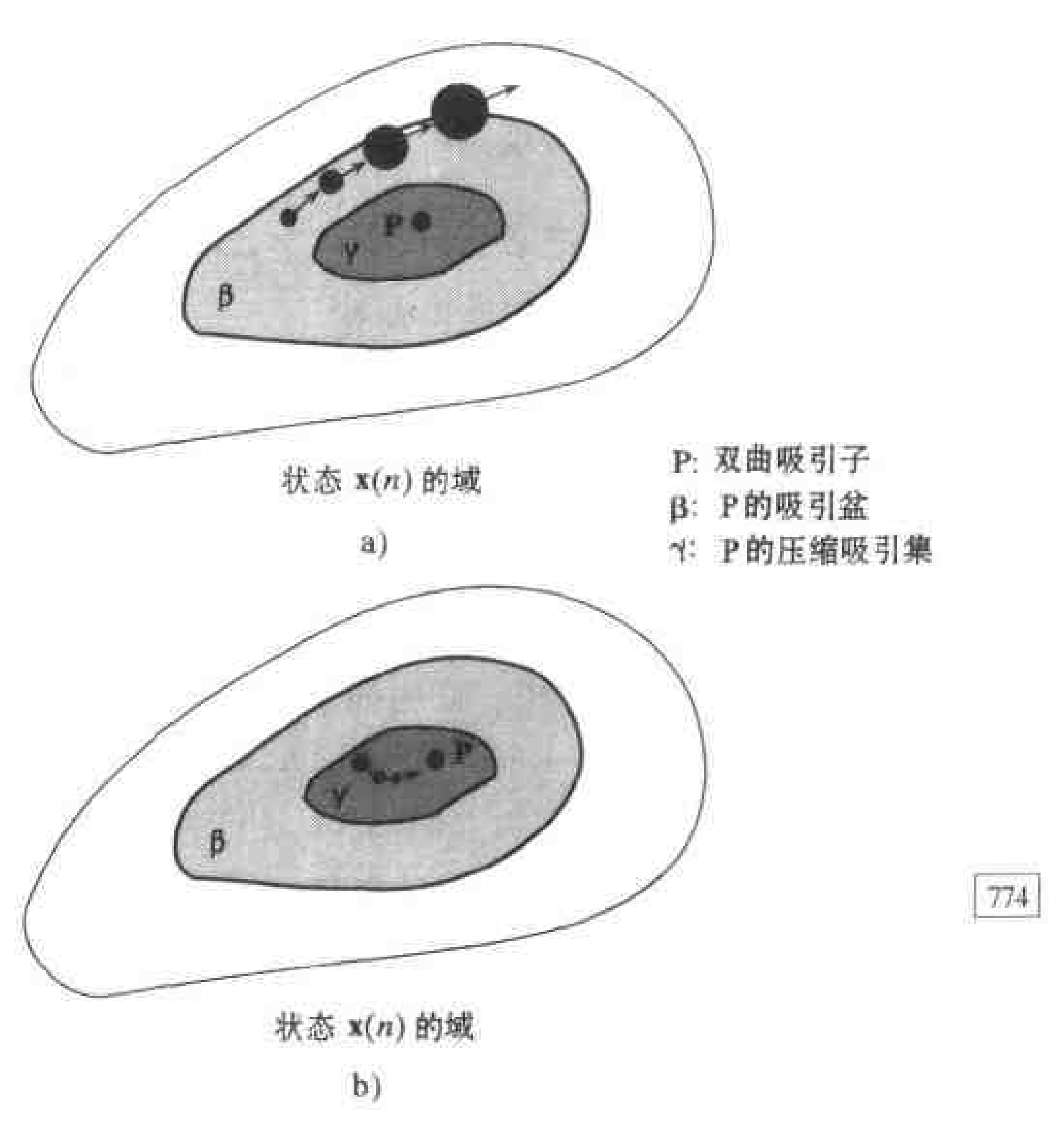


图 15-16
a)状态 $\mathbf{x}(n)$ 在吸引盆 β 内但不在压缩吸引集 γ 内 b)状态 $\mathbf{x}(n)$ 在压缩吸引集 γ 内

[774]

长期依赖

为了理解梯度基础上学习的鲁棒性信息锁存的作用，我们注意在时刻 n 应用到递归网

络的权值向量 $\mathbf{w}$ 由

$$\Delta\mathbf{w}(n)=-\eta\frac{\partial\mathscr{E}_{total}}{\partial\mathbf{w}}$$

调整，这里 η 是学习率参数。$\partial\mathscr{E}_{total}/\partial\mathbf{w}$ 是代价函数 $\mathscr{E}_{total}$ 关于 $\mathbf{w}$ 的梯度。代价函数 $\mathscr{E}_{total}$ 通常由

$$\mathscr{E}_{total}=\frac{1}{2}\sum_i\|\mathbf{d}_i(n)-\mathbf{y}_i(n)\|^2$$

定义，其中 $\mathbf{d}_i(n)$ 是期望响应，$\mathbf{y}_i(n)$ 是网络对第 i 个模式在时间 n 时的实际响应；因此，可以写成的形式：

$$\begin{aligned}\Delta\mathbf{w}(n)&=\eta\sum_i\left(\frac{\partial\mathbf{y}_i(n)}{\partial\mathbf{w}}\right)(\mathbf{d}_i(n)-\mathbf{y}_i(n))\\&=\eta\sum_i\left(\frac{\partial\mathbf{y}_i(n)}{\partial\mathbf{x}_i(n)}\frac{\partial\mathbf{x}_i(n)}{\partial\mathbf{w}}\right)(\mathbf{d}_i(n)-\mathbf{y}_i(n))\end{aligned}\tag{15.81}$$

其中在第二行使用了微积分的链式法则；状态向量 $\mathbf{x}_i(n)$ 属于训练样本的第 i 个模式。在应用诸如通过时间的反向传播算法的时候，代价函数的偏微分根据在不同时间标号的独立权值进行计算。可以扩展方程(15.81)的结果如下：

$$\Delta\mathbf{w}(n)=\eta\sum_i\left(\frac{\partial\mathbf{y}_i(n)}{\partial\mathbf{x}_i(n)}\sum_{k=1}^{n}\frac{\partial\mathbf{x}_i(n)}{\partial\mathbf{w}(k)}\right)(\mathbf{d}_i(n)-\mathbf{y}_i(n))$$

第二次应用微积分的链规则得到

$$\Delta\mathbf{w}=\eta\sum_i\left(\frac{\partial\mathbf{y}_i(n)}{\partial\mathbf{x}_i(n)}\sum_{k=1}^{n}\frac{\partial\mathbf{x}_i(n)}{\partial\mathbf{x}_i(k)}\frac{\partial\mathbf{x}_i(k)}{\partial\mathbf{w}(k)}\right)(\mathbf{d}_i(n)-\mathbf{y}_i(n))\tag{15.82}$$

根据状态方程(15.2)我们认识到有

$$\mathbf{x}_i(n)=\boldsymbol{\varphi}(\mathbf{x}_i(k),\mathbf{u}(n))\qquad 1\leqslant k<n$$

因此我们可以把 $\partial\mathbf{x}_i(n)/\partial\mathbf{x}_i(k)$ 解释为非线性函数 $\boldsymbol{\varphi}(\cdot,\cdot)$ 扩展到 $n-k$ 个时间步的 Jacobi 矩阵，即

775

$$\frac{\partial\mathbf{x}_i(n)}{\partial\mathbf{x}_i(k)}=\frac{\partial\boldsymbol{\varphi}(\mathbf{x}_i(k),\mathbf{u}(n))}{\partial\mathbf{x}_i(k)}=\mathbf{J}_x(n,n-k)\tag{15.83}$$

在Bengio et el.,(1994)中，证明如果输入 $\mathbf{u}(n)$ 使得递归网络在时间 $n=0$ 之后鲁棒地锁存在双曲吸引子内，于是 Jacobi 矩阵 $\mathbf{J}_x(n,k)$ 关于 k 是指数递减的，因此有

$$\det(\mathbf{J}_x(n,k))\to 0\text{ 当 }k\to\infty\text{ 对所有 }n\tag{15.84}$$

式(15.84)的含义是网络的权值向量 $\mathbf{w}$ 的一个微小变化在最近的过去(即接近当前时间 n 的 k 的值)有作用。在时间 n 时可能存在权值向量 $\mathbf{w}$ 的调整 $\Delta\mathbf{w}$ 使得 $\mathbf{x}(n)$ 移动到一个更好的状吸盆，但代价函数 $\mathscr{E}_{total}$ 对 $\mathbf{w}$ 的梯度并不携带那个信息。

作为结论，假设递归网络的双曲吸引子存储状态信息时使用基于梯度的学习，我们可以发现下列两种情况之一：

- 在输入信号具有噪声时网络不是鲁棒的，或者
- 网络不能发现长期性依赖(即时间间隔比较长的输入和目标输出之间的关系)

减轻递归网络中由于消失梯度所产生的困难包括如下可能的过程[13]：

- 在训练过程中，利用基于短符号串优先的原则表示网络以增加输入－输出依赖的时序扩展，参考 15.6 节中的启发方法。

- 用扩展的 Kalman 滤波器或它的解耦形式，比基于梯度的算法更高效地使用可用的信息；扩展的 Kalman 滤波器在 15.10 节讨论。
- 使用诸如拟 Newton 最优化和模拟退火(Bengio et al.,1994)等更精致的优化算法；二阶优化方法和模拟退火分别在第 4 章和第 11 章讨论。

15.13 系统辨识

系统辨识(system identification)是对一个过程或一族未知参数建模的实验方法[14]。它涉及如下步骤：实验计划，选择模型结构，参数估计和模型验证。和实际中所做的一样，系统辨识的过程是迭代性的，我们可能不得不在这些步骤间来回重复直到建立满意的模型为止。

假设已有一个未知的非线性动态设备，需要为它建立合适的参数化的辨识模型。我们选择在状态空间模型或输入-输出模型基础上建立系统辨识过程。决定由哪一个去表示，取决于输入的先验信息和系统的可观测量。下面，对两种表示都进行讨论。

使用状态空间模型的系统辨识

假设给定的设备(plant)由状态空间模型描述：

$$\mathbf{x}(n+1)=\mathbf{f}(\mathbf{x}(n),\mathbf{u}(n)) \tag{15.85}$$

$$\mathbf{y}(n)=\mathbf{h}(\mathbf{x}(n)) \tag{15.86}$$

776

其中这里 $\mathbf{f}(\cdot,\cdot)$ 和 $\mathbf{h}(\cdot)$ 为向量值的非线性函数，两者都假设为未知的；式(15.86)是式(15.11)的一般形式。用两个神经网络去辨识系统，一个处理过程方程(15.85)，另一个处理度量方程(15.86)，如图 15-17 所示。

我们认识到状态 $\mathbf{x}(n)$ 是 $\mathbf{x}(n+1)$ 的单步延迟形式。令 $\hat{\mathbf{x}}(n+1)$ 表示由第一个神经网络产生的 $\mathbf{x}(n+1)$ 的估计，这个神经网络在图 15-17a 中标记为 I。这个网络对包括外部输入 $\mathbf{u}(n)$ 和状态 $\mathbf{x}(n)$ 的并置输入进行操作以产生 $\hat{\mathbf{x}}(n+1)$。从实际状态 $\mathbf{x}(n+1)$ 中减去估计值 $\hat{\mathbf{x}}(n+1)$ 得到误差向量

$$\mathbf{e}_{\mathrm{I}}(n+1)=\mathbf{x}(n+1)-\hat{\mathbf{x}}(n+1)$$

其中 $\mathbf{x}(n+1)$ 起到期望响应的作用。在这个方法中假设状态 $\mathbf{x}(n)$ 实际上是可用的。误差向量 $\mathbf{e}_{\mathrm{I}}(n+1)$ 用作调整神经网络 I 的突触权值，如图 15-17a 所示，所以在统计意义下最小化以误差向量 $\mathbf{e}_{\mathrm{I}}(n+1)$ 为基础的代价函数。

图 15-17b 中标记为 II 的第二个神经网络，通过对未知模型的实际状态 $\mathbf{x}(n)$ 的操作产生实际输出 $\mathbf{y}(n)$ 的估计值 $\hat{\mathbf{y}}(n)$。从 $\mathbf{y}(n)$ 中减去估计值 $\hat{\mathbf{y}}(n)$ 得到第二误差向量

$$\mathbf{e}_{\mathrm{II}}(n)=\mathbf{y}(n)-\hat{\mathbf{y}}(n)$$

其中 $\mathbf{y}(n)$ 起到期望响应的作用。误差向量 $\mathbf{e}_{\mathrm{II}}(n)$ 用于调整网络 II 的突触权值，使得在统计意义下最小化误差向量 $\mathbf{e}_{\mathrm{II}}(n)$ 的欧几里德范数。

图 15-17 所示的两个神经网络在同步模式下运行，提供系统辨识问题的状态空间解(Narendra and Parthasarathy,1990)。考虑到未知系统(而不是辨识模型)的实际状态被反馈到辨识模型这个事实，该模型被称作*串并行辨识模型*(series-parallel identification model)，如图 15-17a 所示。根据 15.9 节最后的讨论，这种形式的训练方法是教师强制的一个例子。

777

图 15-17a 的串并行辨识模型应该与*并行辨识模型*作比较，在后一模型中作用在网络 I

的 $\mathbf{x}(n)$被 $\hat{\mathbf{x}}(n)$代替；其中 $\hat{\mathbf{x}}(n)$是通过传递一个单位延时 $z^{-1}\mathbf{I}$ 从网络自身的输出 $\hat{\mathbf{x}}(n+1)$ 得到的。这个训练替代模型的实际好处是神经网络模型和未知系统运行方式完全相同，也就是说，当训练结束后模型将被使用。因此通过并行训练方式得到的模型比通过串并行方式训练得到的模型有更好的自治行为。但并行训练方式的不利之处在于它的时间比串并行方式时间更长，参考15.9节所讨论的教师强制。特别地，在当前情况下用于并行训练方式的状态估计值 $\hat{\mathbf{x}}(n)$通常不如用于串并行训练方式的实际状态 $\mathbf{x}(n)$准确。

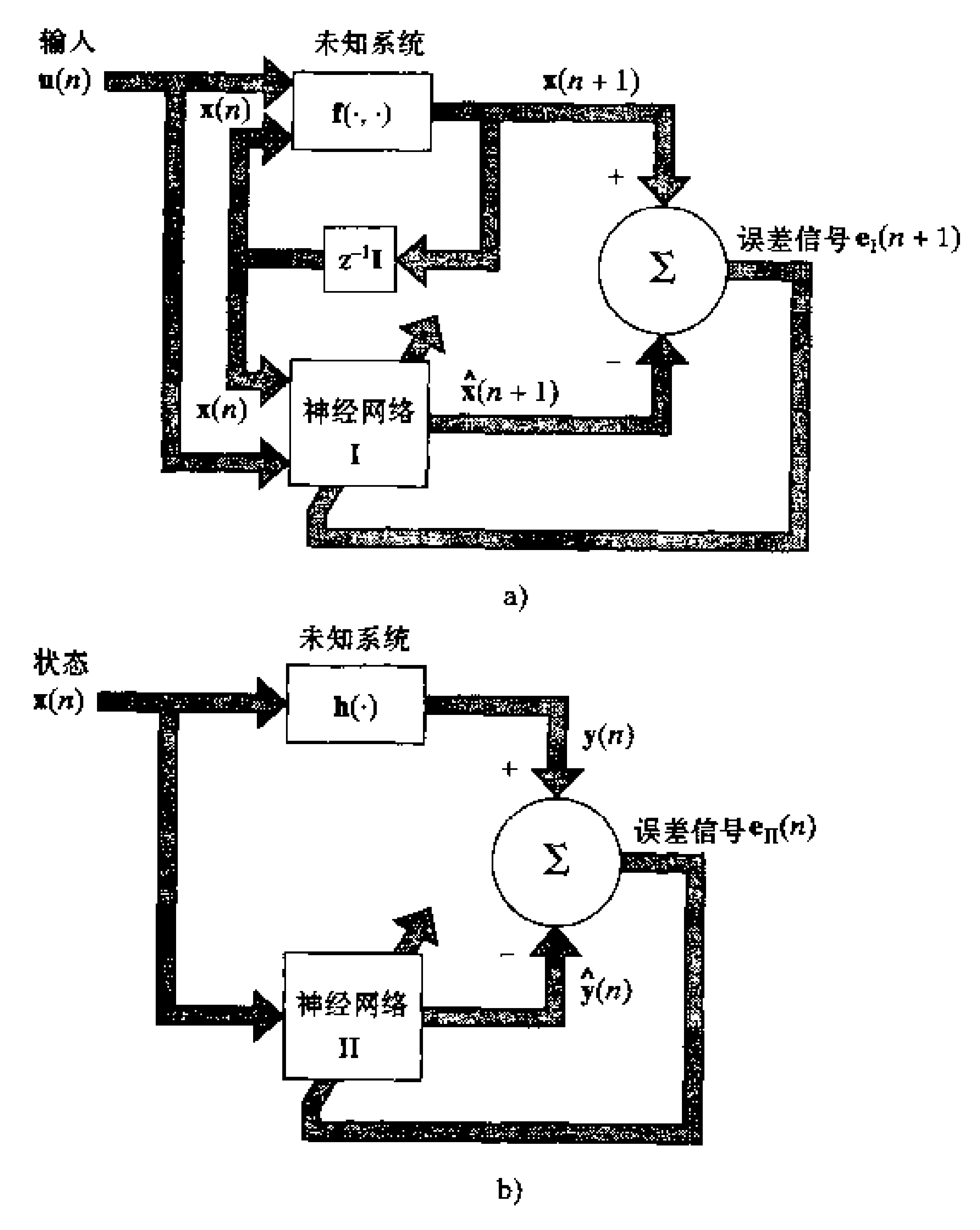


图15-17 系统辨识问题的状态空间解

输入-输出模型

假设下一个未知设备(plant)只能通过它的输出访问。为简化表达，假设系统为单输入单输出的。$y(n)$表示在不同离散时刻 n 时关于输入 $u(n)$的输出。使用NARX模型，辨识模型有如下形式：

$$\hat{y}(n+1)=\varphi(y(n),\cdots,y(n-q+1),u(n),\cdots,u(n-q+1))$$

778 其中 q 是未知系统的阶。在时间 $n+1$，输入的 q 个过去值和输出的 q 个过去值都可用。模型输出 $\hat{y}(n+1)$表示实际输出 $y(n+1)$的估计值。从 $y(n+1)$中减掉估计 $\hat{y}(n+1)$得到误差信号

$$e(n+1) = y(n+1) - \hat{y}(n+1)$$

其中 $y(n+1)$起着期望响应的作用。利用误差 $e(n+1)$调整神经网络的突触权值使得在统计意义下最小化误差。因为系统(而不是辨识模型)的实际输出被反馈回模型的输入，如图 15-18 的辨识模型是一个串并行形式(即教师强制形式)。

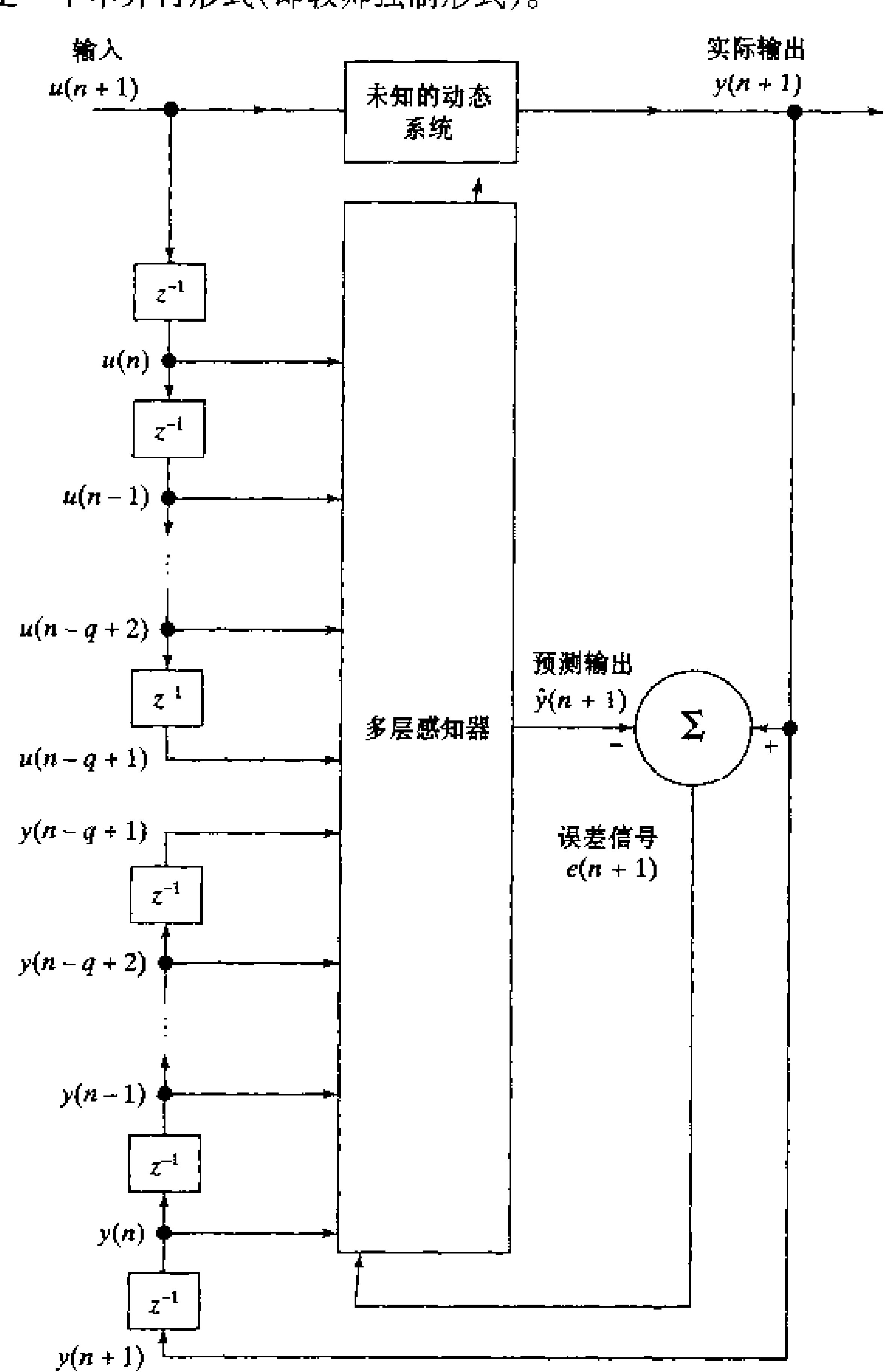


图 15-18 系统辨识问题的 NARX 解

15.14 模型参考自适应控制

递归网络的另一应用是设计反馈控制系统，在这里设备(plant)状态由强加的控制非线性地耦合(Puskorius and Feldkamp, 1994; Puskorius et al., 1996)。系统的设计由其他因素例如无法测量的随机扰动、可能系统的逆不惟一以及出现不可观察的系统状态而进一步复杂化。

适合使用神经网络的控制策略是*模型参考自适应控制*(model reference adaptive control, MRAC)[115]，这里蕴含的假设是设计者对所考虑的系统足够熟悉(Narendra and Annaswany, 1989)。图15-19显示这样一个系统的框图，其中自适应性用来解释系统的动力学性质是未知的这个事实。控制器和系统形成一个封闭的环状反馈系统，因此组成一个*外部回归*(externally recurrent)网络。设备从控制器接受输入 $\mathbf{u}_c(n)$ 以及外部的一个扰动 $\mathbf{u}_d(n)$。相应地，设备及时地演化为强制输入和系统自身状态 $\mathbf{x}_p(n)$ 的函数。设备输出 $\mathbf{y}_p(n+1)$ 是 $\mathbf{x}_p(n)$ 的函数。设备输出也可能被度量噪声所损坏。

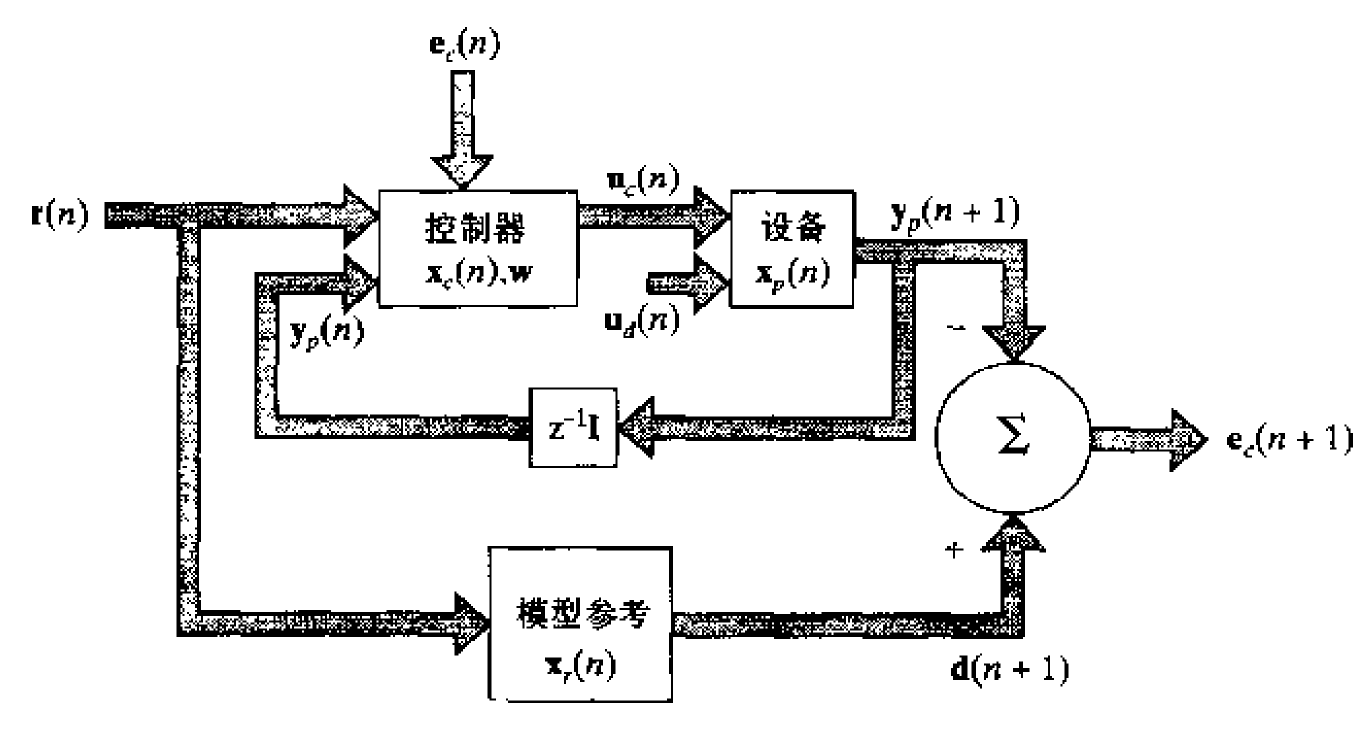


图15-19 使用直接控制的模型参考自适应控制

780

控制器接受两个输入：外部指定的参考信号 $\mathbf{r}(n)$，以及表示设备输出 $\mathbf{y}_p(n+1)$ 单步延迟形式的 $\mathbf{y}_p(n)$。控制器产生控制信号向量，定义为

$$\mathbf{u}_c(n) = \mathbf{f}_1(\mathbf{x}_c(n), \mathbf{y}_p(n), \mathbf{r}(n), \mathbf{w})$$

其中 $\mathbf{x}_c(n)$ 为控制器自身的状态，$\mathbf{w}$ 是可调的参数向量。向量值函数 $\mathbf{f}_1(\cdot,\cdot,\cdot,\cdot)$ 定义控制器的输入-输出行为。

设备期望响应 $\mathbf{d}(n+1)$ 是由稳定*参考模型*(reference model)的输出提供的，它是响应参考 $\mathbf{r}(n)$ 而产生的。期望响应 $\mathbf{d}(n+1)$ 因此是参考信号 $\mathbf{r}(n)$ 和参考模型自身状态 $\mathbf{x}_r(n)$ 的函数，表示为

$$\mathbf{d}(n+1) = \mathbf{f}_2(\mathbf{x}_r(n), \mathbf{r}(n))$$

向量值函数 $\mathbf{f}_2(\cdot,\cdot)$ 定义参考模型的输入-输出行为。

输出误差(即设备和模型参考输出之间的误差)记为

$$\mathbf{e}_c(n+1) = \mathbf{d}(n+1) - \mathbf{y}_p(n+1)$$

设计目标是调整控制器的参数向量 $\mathbf{w}$，使得输出误差 $\mathbf{e}_c(n)$ 的欧几里德范数是对时间 n 的最小化。

图15-19的MRAC系统的控制方法被称为*直接的*，这是指不用辨识设备参数，而是直接调整控制器的参数提高系统性能。不幸的是，当前还没有在输出误差基础上调整控制器参数的精确方法(Narendra and Parthasarathy, 1990)。这是因为未知设备处于控制器和输出误差之间。为克服这个困难，我们可以用*间接控制*(indirect control)，如图15-20所示。后面这种方法，使用两步过程训练控制器：

1. 设备 P 的模型记为 $\hat{P}$，它是根据系统输出对设备输入、以前的设备输出和以前的设备内部状态的微分关系的估计而得到的。在前一节描述的过程用于训练神经网络使之辨识设备；这样得到模型 $\hat{P}$ 称为*辨识模型*。 781
2. 使用辨识模型 $\hat{P}$ 替代设备以得到设备输出对控制器可调整参数向量的动态导数的估计。

在间接控制里，*外部递归网络*包括控制器和通过辨识模型 $\hat{P}$ 产生的设备输入/输出表示。

在图 15-20 的一般结构中，递归网络对于控制器设计的应用有一系列广为人知例子，如小车－单立摆(cart-pole)问题，生物反应器标准测试(bioreactor benchmark)问题以及自动控制子系统，即发动机慢速(engine idle-speed)控制(Puskorius and Feldkamp, 1994, Puskorius et al., 1996)。在这些研究里的递归网络是和 15.2 节的讨论相似的递归多层感知器。网络的训练使用 15.11 节的 DEKF 算法。但注意，对于发动机慢速控制因为强加的控制(在适当选择的范围内)单调地影响发动机速度，选择了一个线性动态系统作为辨识模型。

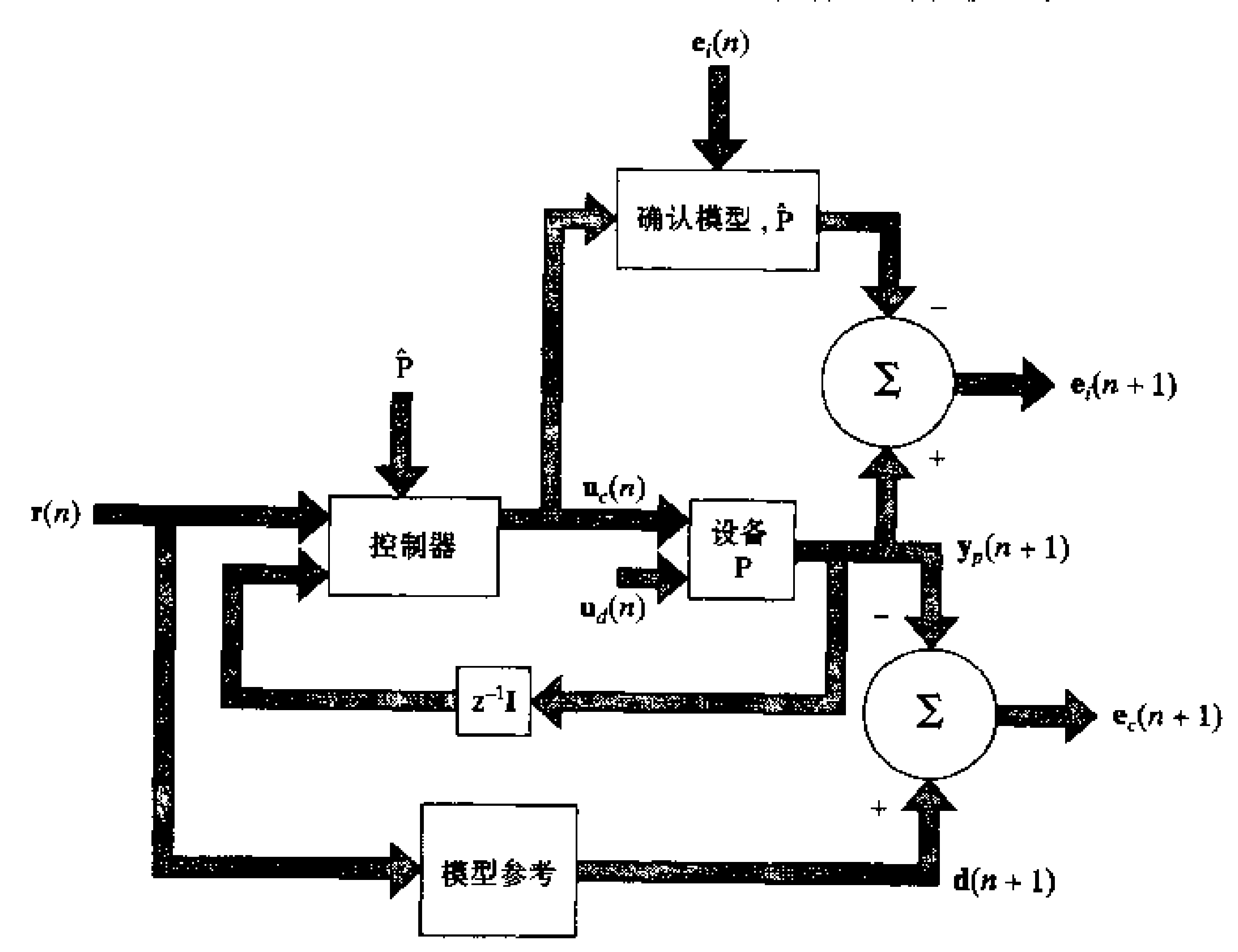


图 15-20 通过辨识模型利用间接控制的模型参考自适应控制

15.15 小结和讨论

本章讨论涉及应用*全局反馈*到静态(无记忆)多层感知器的递归网络。反馈的应用使得神经网络获得状态表示，使得它们成为信号处理和控制中各种应用的合适工具。属于有全局反馈的递归网络类型的四个主要网络结构如下：

- 使用从输出层反馈到输入层的具有外部输入的非线性自回归(NARX)网络。
- 具有从隐藏层到输入层反馈的完全连接递归网络。
- 有多于一个隐藏层的递归多层感知器，其中每个计算层输出反馈到它自己的输入。

- 使用二阶神经元的二阶递归网络。

在所有这些递归网络中，反馈通过抽头延迟线记忆。

前三个递归网络可以使用状态空间框架研究其动态行为。这个根植于现代控制论的方法提供一个研究非线性动态递归网络的一个有力的工具。

我们描述三种基本的算法来训练递归网络的算法：通过时间的反向传播(BPTT)，实时递归学习(RTRL)，和解耦扩展的 Kalman 滤波器(DEKF)。BPTT 和 RTRL 算法是建立在梯度基础上的，而 DEKF 算法对高阶信息的使用更有效。因此它可以比 BPTT 和 RTRL 收敛更快，但也增加相应的计算复杂性。实际上 DEKF 算法可以看做是一种可能使用的技术，它使得解决困难的信号处理和控制问题成为可能。

782

理论上，有全局反馈(例如使用 DEKF 算法训练的递归多层感知器)的递归网络可以学习非定常(nonstationary)环境下的固有动力学系统，这是通过将从训练样本中获得的知识存储在一个固定的权值集合中实现的。更重要的是，假设满足下面两个条件网络可以追踪环境的统计变化：

- 递归网络不发生欠适应(underfitting)或过适应(overfitting)。
- 训练样本表示环境的非定常行为。

综观全章，我们强调利用递归网络进行时序处理。递归网络也可以用于处理一系列有序的数据，这些数据并没有直接的时序解释(如表示为树的化学结构)。在 Sperduti and Starita (1997)中，递归网络可以表示和分类结构化模式，这些模式可以表示成有向图、带标号图和无环图的形式。这种方法背后的主导思想是在这里被称作“广义递归神经元”，这是指一个递归神经元(即具有局部反馈的神经元)结构上的推广。通过使用这样一个模型，监督学习算法诸如通过时间的反向传播和实时递归学习都可以被扩展以处理结构化模式。

注释和参考文献

[1] 关于其他递归网络结构，见 Jordan(1986)，Back and Tsoi(1991)，Frasconi et al.，(1992)，以及 Robinson and Fallside(1991)。

[2] NARX 模型包括一类重要的非线性离散时间系统(Leontaritis and Billings,1985)。涉及到神经网络这方面的讨论可以参考 Chen et al.，(1990)，Narendra and Parthasarathy(1990)，Lin et al.，(1996)和 Sieglemann et al.，(1997)。

已经证实 NARX 模型十分适合对非线性系统进行建模，如热交换器(Chen el al.,1990)，污水处理设备(Su and McAvoy,1991;Su et al.,1992)，用于石油提炼的催化更新系统(Su et al.,1992)，在生物系统中的多肢移动的非线性振荡(Venkataraman,1994)和语法推理(Giles and Horne,1994)。

NARX 模型也指非线性自回归滑动平均(NARMA)模型，其中“滑动平均”是对于输入而言。

[3] 图 15-4 的递归多层感知器是 Jordan(1986)描述的递归网络的推广。

[4] Omlin and Giles(1996)指出，用二阶递归网络，任何有限状态自动机可以映射到这样一种网络，且可以保证有限长度的时序序列的正确分类。

[5] 可控性和可观察性的严格处理可以参考 Zadeh and Desoer(1963)，Kailath(1980)，Sontag(1990)，Lewis and Syrmos(1995)。

[6] 有关神经网络和自动机(实际上是串行机器 - 自动机的实现)方面的最早工作，即第一篇关于有限状态自动机、人工智能和递归神经网络方面的论文，是 McCulloch and Pitts (1943)的著名的论文。递归网络(具有瞬时反馈)是这篇论文的第二部分，这在 Kleene (1956)被解释为一个有限状态自动机。Kleene 的论文出现在由 Shannon 和 McCarthy 编辑的《自动机研究》(Automata Studies)一书中(这本惊世之作的作者还包括 Moore，Minsky，von Neumann，Uttley，McCarthy 和 Shannon 等人)。有时候，Kleene 的论文被作为有限状态机器方面的第一篇文章引用(Perrin，1990)。Minsky(1967)在他的《计算：有限和无限机器》(Computation：Finite and Infinite Machines)一书中讨论自动机和神经网络。 783

所有关于自动机和神经网络方面的早期工作主要考虑怎样将二者结合在一起，就是说，如何建造和设计自动机到神经网络中去。因为大多数自动机(当被实现为串行机器的时候)需要反馈，神经网络必须为递归的。注意早期的工作(除了 Minsky 的)并没有明确地区分自动机(有向图，标记图，无圈图)和串行机器(逻辑延时和反馈延时)，大多数情况下仅考虑有限状态自动机。对于提高自动机的层次到下推自动机和图灵机没有什么兴趣(除了 Minsky 之外)。

在神经网络的黑暗时代过去之后，关于自动机和神经网络方面的研究在 20 世纪 80 年代又开始了。这个工作可以大概分为下面三个大的领域：(1) 学习自动机，(2)自动机关于知识的合成、抽取和提炼，(3)表示。首先提到自动机和神经网络的是 Jordan (1986)。

[7] 使用 McCulloch - Pitts 神经元的单层递归网络不能模拟任何有限状态的机(Goudreau et al.，1994)，但 Elman 的简单递归网络可以作这样的模拟(Kremer，1995)。只有局部反馈的递归网络不能表示所有有限状态机(Frasconi and Gori，1996；Giles et al.，1995；Kremer，1996)。

[8] 通过时间的反向传播的思想，是对于每一个递归网络都可能建立一个前馈网络，使之在一个特定的时间间隔内具有和它相同的行为(Minsky and Papert，1969)。通过时间的反向传播首先由 Werbos(1974)的博士论文讨论；也可以参考 Werbos(1990)。这个算法由 Rumelhart et al.，(1986b)独立地重新发现。通过时间的反向传播算法的一个变体由 Williams and Peng (1990)所讨论。对于算法的综述和相关的问题，可以参考 Williams and Zipser(1995)。

[9] 实时递归学习算法在神经元网络文献中的第一次描述是 Williams and Zipser(1989)。其来源可以追溯到 McBride and Narendra(1965)用于调节任意动态系统参数的系统辨识的论文。

Williams 和 Zipser 给出的推导是关于完全递归的单层神经网络。它已扩展为更一般的结构；例如，参考 Kechriotis et al.，(1994)；Puskorius and Feldkamp(1994)。

[10] Kalman 滤波器理论来源于 Rudolf E. Kalman(1960)的经典论文。它已成为信号处理和控制的核心部分，并且在很多领域有很广泛的应用。对于标准 Kalman 滤波器、它的变体和它的用于处理非线性动态系统的扩展形式以及它们的详细细节，可以参考 Grewal and Andrews (1993)和 Haykin(1996)。由 Grewal 和 Andrews 写的书全部讨论的是 Kalman 滤波器的理论和实践。由 Haykin 写的书，从自适应的滤波方面讨论 Kalman 滤波器的理论。另外两本这个方面的重要的书是 Jazwinski(1970)和 Maybeck(1979，1982)。

[11] 平方根 Kalman 滤波器细节处理和实现它的有效方法，见 Haykin(1996)。

[12] Singhal and Wu(1989)也许是第一个展示用扩展的 Kalman 滤波器提高监督神经元网络的映射性能。不幸的是，那里讨论的训练算法受限于它的计算的复杂性。为克服这个困难，Kollias and Anastassiou(1989)，Shah and Palmieri(1990)尝试通过将全局问题分为一系列子问题，每个子问题表示一个单一的神经元，以简化扩展的 Kalman 滤波器的应用。但是作为一个辨识问题的每一个神经元的处理并不是严格地遵守 Kalman 滤波器理论。还有，这样处理会导致训练过程中的不稳定行为，并且可能得到比别的方法得到的结果还差的解(Puskorius and Feldkamp,1991)。

784

[13] 消失梯度问题的其他处理方法包括绕过一些递归网络的非线性特性以便改进长期学习的依赖性。这种处理的例子包括：

- 在网络体系结构中使用长期延迟(El Hihi and Bengio,1996;Lin et al.,1996;Giles et al.,1997)
- 与不同时间尺度联系的多级网络层次化结构(El Hihi and Bengio,1996)
- 用门单元避开某些非线性性性(Hochreiter and Schmidhuber,1997)

[14] 系统辨识有许多文献。对于这个主题讨论的书籍，可以参考 Ljung(1987)，Ljung and Glad (1994)。对于这个问题特别是将重点集中在神经网络上的综述可以参考 Sjöberg et al.，(1995)和 Narendra(1995)。使用神经网络对于系统辨识进行详细的研究首先是 Narendra and Parthasarathy(1990)。

[15] 对模型参考自适应控制的详细讨论，见 Landau(1979)的书。

习题

状态空间模型

15.1 写出图 15-3 的 Elman 简单递归网络状态空间模型的计算公式。

15.2 证实图 15-4 的递归多层感知器可以用状态空间模型

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n), \mathbf{u}(n))$$
$$\mathbf{y}(n) = \mathbf{g}(\mathbf{x}(n), \mathbf{u}(n))$$

表示，其中 $\mathbf{u}(n)$表示输入，$\mathbf{y}(n)$表示输出，$\mathbf{x}(n)$表示状态，$\mathbf{f}(\cdot,\cdot)$和 $\mathbf{g}(\cdot,\cdot)$表示向量值非线性函数。

15.3 一个动态系统是否可能是可控的但不可观察的，而且反之亦然？证实你的答案。

15.4 参考 15.3 节的局部可控性问题，证实

(a)状态 $\mathbf{x}(n+q)$是它过去值 $\mathbf{x}(n)$和式(15.24)的输入向量 $\mathbf{u}_q(n)$的嵌套非线性函数。

(b)$\mathbf{x}(n+q)$对 $\mathbf{u}_q(n)$的 Jacobi 矩阵在原点求值等于式(15.23)可控性矩阵 $\mathbf{M}_c$。

15.5 参照 15.3 节的局部可观察性问题，证明定义在式(15.30)中的观察向量 $\mathbf{y}_q(n)$对状态 $\mathbf{x}(n)$的 Jacobi 矩阵在原点的求值等于式(15.28)的可观察矩阵 $\mathbf{M}_o$。

15.6 非线性动态系统的过程方程由

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{x}(n), \mathbf{u}(n))$$

785

描述，其中 $\mathbf{u}(n)$是在时刻 n 的输入向量，$\mathbf{x}(n)$是对应的系统状态。输入 $\mathbf{u}(n)$过程方程中以非加性的方式出现。在本题中，我们希望重新写过程方程，使输入 $\mathbf{u}(n)$以加性的方式出

现。这仅需写成

$$\mathbf{x}'(n+1)=\mathbf{f}_{new}(\mathbf{x}'(n))+\mathbf{u}'(n)$$

给出向量 $\mathbf{x}'(n)$和 $\mathbf{u}'(n)$以及函数 $\mathbf{f}_{new}(\cdot)$的定义公式。

15.7 图 15-22 提出在神经元级上的使用局部反馈的递归网络模型的两个例子。在图中的 a 部分和 b 部分显示的体系结构分别称为*局部激活反馈*和*局部输出反馈*(Tsoi and Back, 1994)。对这两个递归网络的体系结构，写出状态空间模型公式。评价它们的可控性和可观察性。

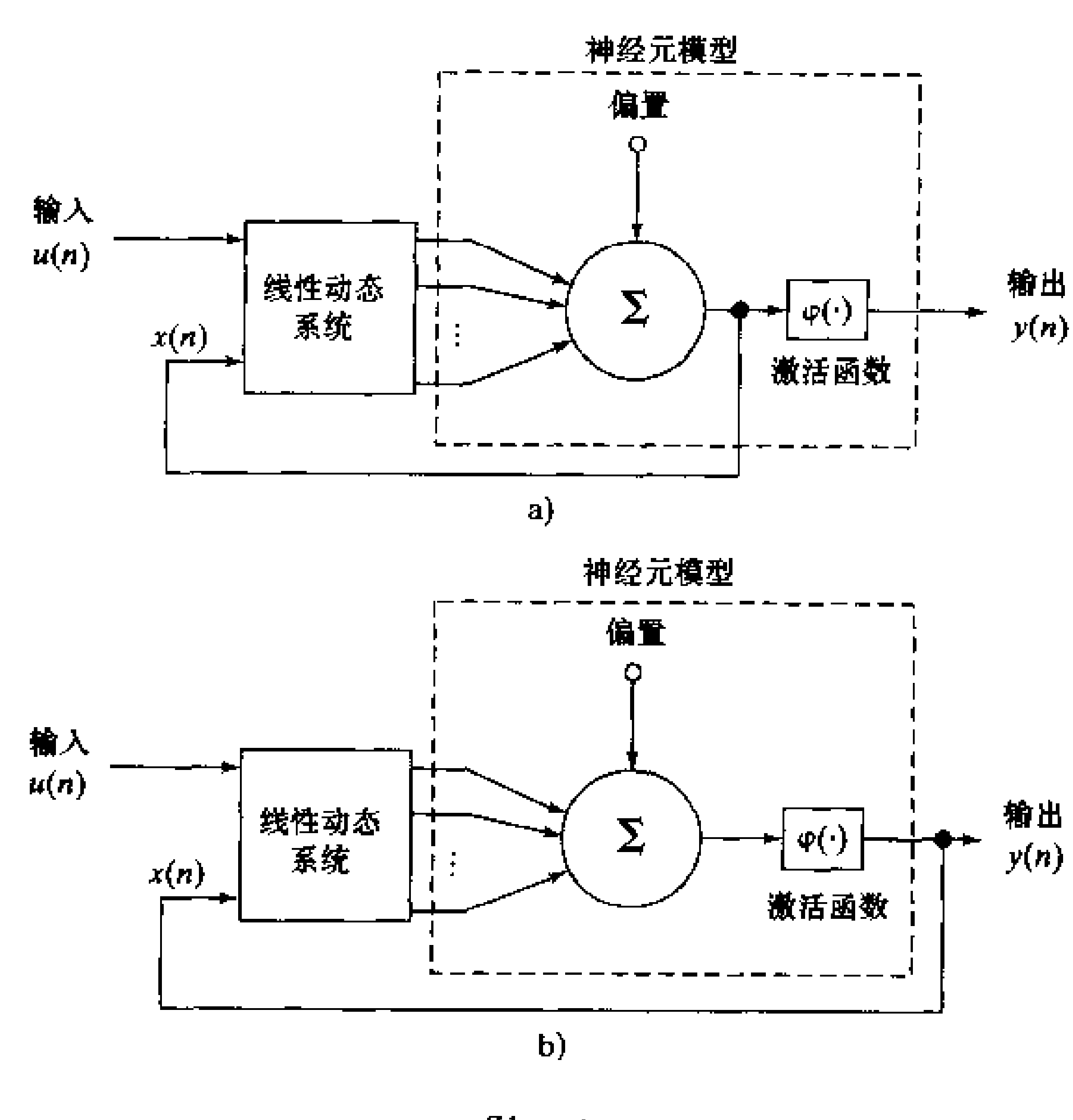


图 15-21

a)局部激活反馈体系结构 b)局部输出反馈体系结构

有外部输入的非线性自回归(NARX)模型

15.8 参考 15.4 节的 NARX 模型，证明式(15.16)和(15.17)的使用导致 NARX 模型的输出 $y(n+q)$关于状态 $\mathbf{x}(n)$和输入向量 $\mathbf{u}_q(n)$的表达如下：

$$y(n+q)=\Phi(\mathbf{x}(n),\mathbf{u}_q(n))$$

其中 Φ: $\mathbb{R}^{2q}\to\mathbb{R}$，$\mathbf{u}_q$ 按式(15.29)定义。

15.9 (a)15.4 节讨论的 NARX 模型的推导是单输入单输出系统。讨论那里描述的理论如何推广到多输入多输出系统。 786

(b)建立等价于图 15.6 中的两个输入一个输出的状态空间模型的 NARX。

15.10 建立对应于图 15-22 中的完全递归网络的 NARX。

15.11 在 15.4 节我们证明了任何状态空间模型可以表达成 NARX 模型。反过来的结果如何？任何的 NRAX 模型是否都可以表达成 15.3 节形式的状态空间模型？说明你的结论的理由。

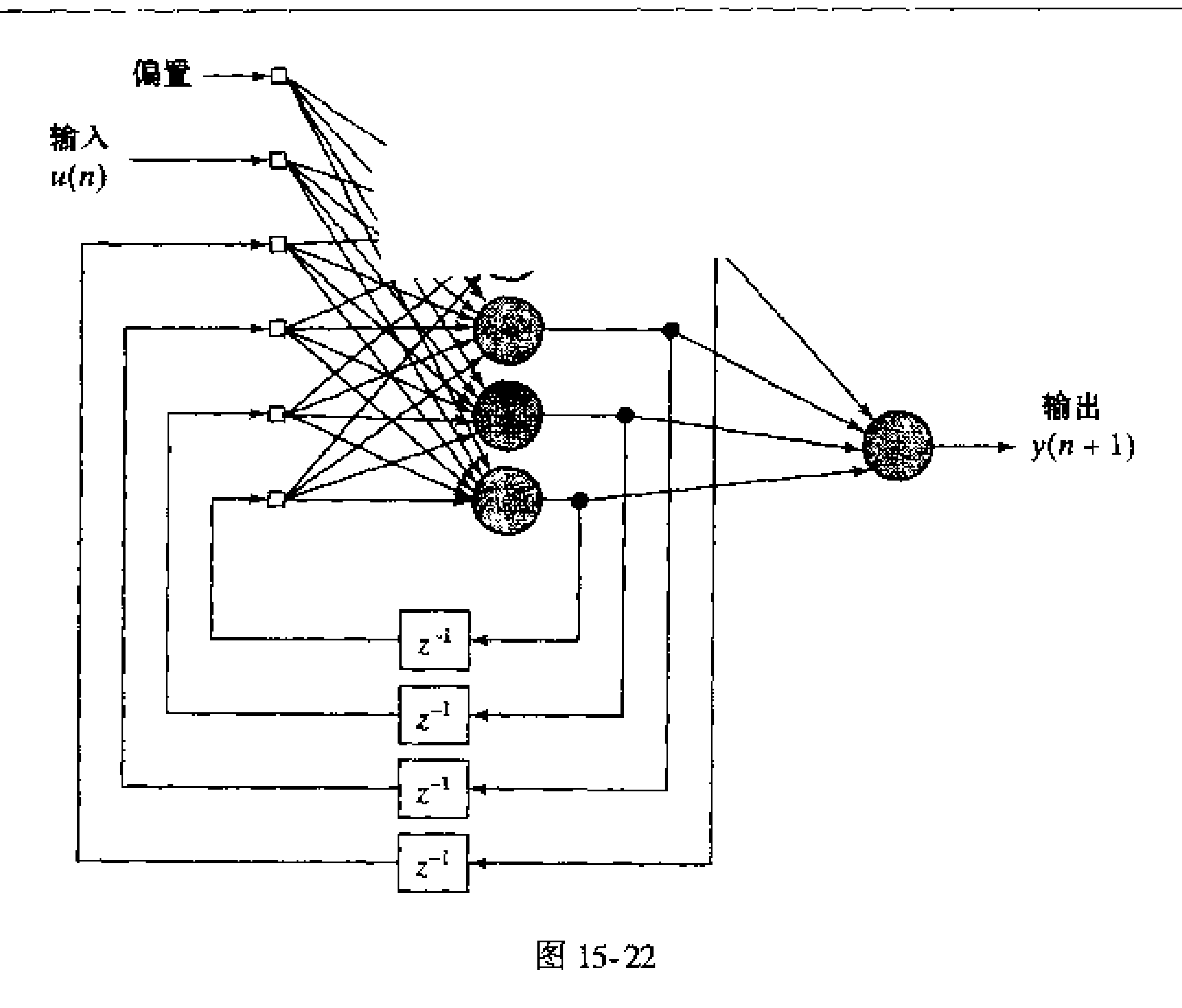


图 15-22

通过时间的反向传播

15.12 展开图 15-3 的状态空间模型的时序行为。

15.13 截断的 BPTT(h)算法可以看作是分回合的 BPTT 算法的近似。可以通过将分回合 BPTT 算法的一些方面包括进 BPTT(h)来提高这个近似程度。特别是可以让网络在执行下一个 BPTT 计算前通过 h'个附加步，这里 $h' < h$。通过时间的反向传播的混合形式的重要特征是下一个后向传播在时间步 $n + h'$之后才执行。在此期间，网络过去输入值、网络状态和期望的响应都存储在一个缓冲区里面，但并不对于它们进行处理(Williams and Peng, 1990)。在这个混合型的算法中给出神经元 j 的局部梯度的公式。

实时递归学习算法

15.14 教师强制递归网络在训练过程中的动态在 15.8 节中描述，但是要除开下面的变化：

787

$$\xi_i(n) = \begin{cases} u_i(n), & \text{如果 } i \in \mathcal{A} \\ d_i(n), & \text{如果 } i \in \mathcal{C} \\ y_i(n), & \text{如果 } i \in \mathcal{B} - \mathcal{C} \end{cases}$$

其中$\mathcal{A}$是当 ξ_i 是一个外部输入时下标为 i 的集合。$\mathcal{B}$表示当 ξ_i 是一个神经元的输出时下标 i 的集合，$\mathcal{C}$表示可见的输出神经元的集合。

(a)证明对这个格式，偏导数$\partial y_j(n+1)/\partial w_{kl}(n)$由下式给出(Williams and Zipser, 1989)：

$$\frac{\partial y_j(n+1)}{\partial w_{kl}(n)} = \varphi'(v_j(n))\left[\sum_{i \in \mathcal{B}-\mathcal{C}} w_{ji}(n)\frac{\partial y_i(n)}{\partial w_{kl}(n)} + \delta_{kj}\xi_l(n)\right]$$

(b)对于教师强制递归网络推导训练算法。

解耦扩展的 Kalman 滤波器(DEKF)算法

15.15 描述图 15-3 的 DEKF 算法如何训练简单递归网络。对于这个训练也可用 BPTT 算

法。

15.16 用通常的形式，DEKF 被用作执行权值更新，一个例子接一个例子的方式进行。反之，在标准反向传播里，执行简单的梯度更新，这使我们可以选择立即使用这些更新还是将这些更新积累一段时间，然后将它们作为单一的组合更新。虽然可以在 DEKF 算法中尝试积累，但这样做也有可能在权值向量和误差协方差矩阵间造成不一致，该矩阵是每个时间递归都更新一次，以产生一个权值更新。DEKF 训练算法的使用表现为排除集中式更新。但可以使用多流(multistream)DEKF 训练，它允许多个训练序列的进行，又保持与 Kalman 滤波器理论的一致性，Feldkamp et al.，(1997)，Feldkamp and Puskorius(1998)中的描述。

(a)考虑有 N_{in}个输入和 N_{out}个输出和固定 N 个训练样本的训练问题。对训练样本来说，组成 $M \leqslant N$ 个数据流以馈给 M 个网络，这些网络受到具有相同权值的限制。在每个训练循环，每个数据流中的模式呈现给各自的网络，对于每个数据流计算出 N_{out}个输出。然后计算单个权值更新并以同样的方式又应用到每个流的网络。推导出 DEKF 算法的多流的形式。

(b)考虑标准 XOR 问题的四种训练模式。假设有一个连接到输出层的延迟线记忆的前馈网络。我们有效地使用四个网络输出：反馈到延迟线记忆的实际的网络输出，三个它的延迟形式，它们中的每一个组成一个新的网络输出。对这个网络结构以一定的顺序应用四种训练模式，但不执行权值更新。当第四个训练模式结束后，就有了四个代表四种训练模式处理过程的网络输出，这是在具有相同权值的网络上进行的。如果考虑在四种训练模式和四个网络输出的基础上执行 DEKF 算法的单一权值向量更新，就有了四个流问题。检查该实例。

二阶递归网络

15.17 在本题中，研究用二阶递归网络建立相似的有限状态自动机。在任意长度的 0，1 序列中，这个自动机可以识别奇数个 1。

788

图 15-23 显示两种状态的自动机。状态由圆圈表示，箭头表示状态的转变。S 表示我们在那个状态开始，在这里是状态 A。粗圆圈表示无论何时达到了那个状态，如图中的状态 B，我们就接受该字符串。自动机开始检查状态 A 的字符串，如果遇到一个 0 就回到状态 A，如果是 1 则回到状态 B。相似地，当在状态 B 的时候，如果遇到一个 0 就回到状态 B，如果遇到 1 则回到状态 A。以这种方式，如果有偶数个 1(包括 0 个)则自动机在状态 A，如果有奇数个 1 则在状态 B。

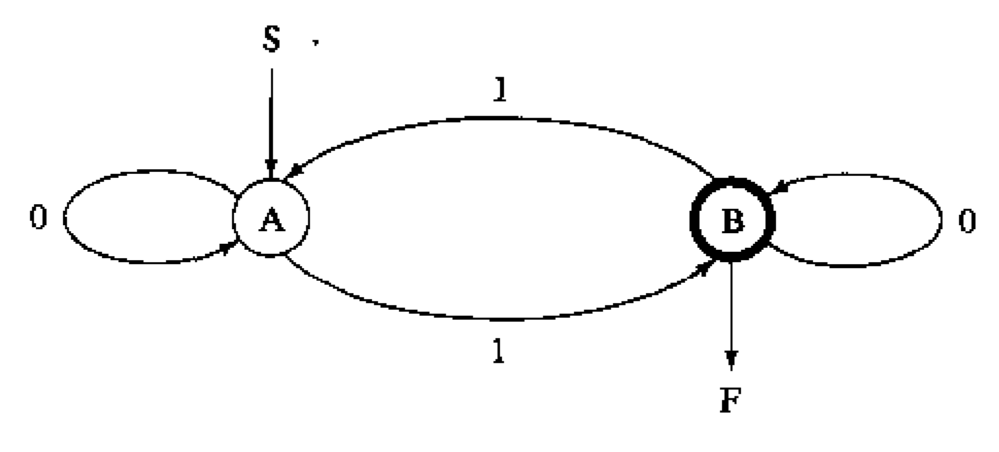


图 15-23

更正式地定义状态 $Q=\{A,\ B\}$，$S=A$ 为初始状态，输入字母为 $\sum=\{0,\ 1\}$，接受状态为 $F=B$，状态转换函数如下：

$$\delta(A,0) = A$$
$$\delta(A,1) = B$$
$$\delta(B,0) = B$$
$$\delta(B,1) = A$$

对于二阶递归网络，这就是式(15.9)的应用需要的一些等式。关于有限状态自动机的细节，见 Hopcroft(1979)。

对上述转换规则进行编码到二阶递归网络中去。

789

15.18 在 15.8 节，我们导出使用一阶神经元的完全连接递归网络的实时递归学习(RTRL)算法。在 15.2 节，我们描述使用二阶神经元的递归网络。

通过推导用于训练二阶递归网络的 RTRL 算法，推广 15.8 节描述的理论。

后　记

神经网络代表一种多学科主题，它植根于神经科学、数学、统计学、物理学、计算机科学和工程学，这可由这本书所涵盖题材的多样性为证。它们在有教师或无教师情况下从数据中学习的能力赋予它们强有力的性质。这种学习性质具有深远的理论和实际意义。神经网络以这种或那种形式从例子(它们环境的表示)学习的能力，已经使得它们在如此众多的应用中成为非常宝贵的工具，比如建模、时间序列分析、模式识别、信号处理和控制。特别地，当一个感兴趣的问题的解由于以下一点或几点变得困难时，神经网络可提供大量的东西：

- 缺乏问题的物理/统计的理解。
- 在可观察数据中的统计变化。
- 数据产生的非线性机制。

神经网络的新浪潮(从20世纪80年代中期开始)已经来临，因为学习可以在许多层次进行。基于学习算法的神经网络使我们可以在手写体识别器中免除手工特征提取。由神经网络激发的基于梯度的学习算法允许我们同时训练特征提取器、分类器和上下文处理器(隐Markov模型和语言模型)。由于神经网络我们学会了从像素到符号的所有途径。

学习渗透到数目日益增加的各种应用智能机器的每个层面。因此，这篇后记以对某些智能机器和神经网络在建立它们时的作用的最终评论结束全书是适宜的。 790

智能机器

由于智能[1]的科学定义尚不统一并且篇幅有限，我们不冒险讨论智能是什么。相反，我们将我们对智能机器的简要解释限制在三个具体应用领域的背景下：模式分类、控制和信号处理。这里要认识到没有“通用的”智能机器；相反，我们只是有针对具体应用的智能机器。

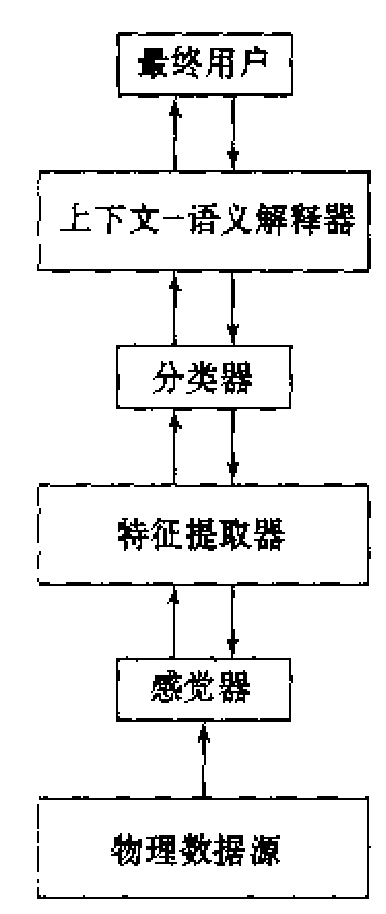


图1　用于模式分类的智能机器的功能结构

神经网络的大部分研究工作集中于模式分类。由于模式分类的实际重要性和它的相当广泛性，以及神经网络如此适于解决模式分类任务的事实，研究努力的这种集中确实是应该的。这样做我们已经能够为自适应模式分类打下基础。但是，我们已经到达另一个阶段，如果希望成功解决更加复杂和困难的模式分类问题，我们必须在一种更广泛的意义上思考分类系统。图1描绘“假定的”分类系统布局(Hammerstrom and Rahfuss, 1992)。系统的第一层接受由信息源产生的感觉数据。第二层提取刻画感觉数据的一组特征。第三层将特征分类为一个或几个不相同的类，然后由第四层将它放入全局背景中。最后，例如，对最终用户我们可能将分析后的输入放入某种数据库形式中。刻画图1系统的重要特征包括： 791

- 识别，起因于信息从系统的一层前向流动到下一层，这如同在传统的模式分类系统

一样

- 集中，凭借系统较高层能够选择性地影响较低层的信息处理，这要依靠从过去数据获得的知识

因此图 1 中显示的模式分类系统的新颖性在于目标领域的知识，以及在给定有限信息处理能力的基本约束下，它被系统较低层利用以便提高整体系统性能。我们相信使用神经网络的模式分类的演化必将沿着创建模型的方向进行，这种模型将持续受到目标领域知识的影响。我们设想用于模式分类的新一类智能机器将提供如下属性：

- 提取背景知识的能力，并且通过集中(focusing)的使用利用这种能力
- 知识的局部化表示而不是分布式表示
- 稀疏结构，强调网络的模块性和层次性作为神经网络设计的原则

这样一种智能机器的实现只有依靠组合神经网络和其他合适工具才有可能得到。这里想到的一个有用工具是 Viterbi 算法，它是动态规划的一种形式，设计用于对付串行信息处理[2]，这种处理是图 1 中描述的系统的固有特征。(动态规划算法在第 12 章讨论。)

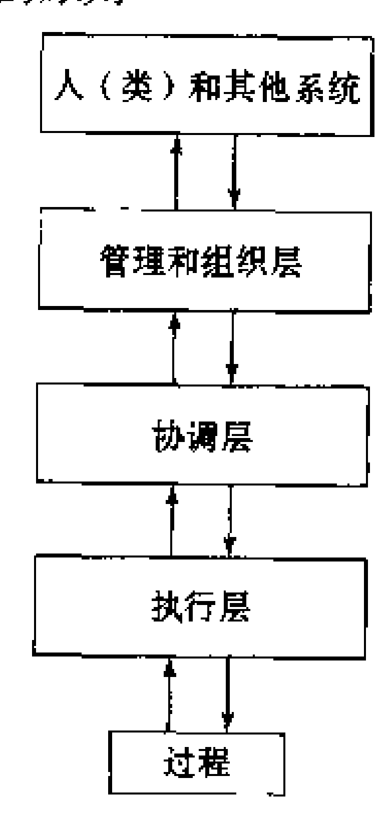


图 2 用于控制的智能机器功能结构

另一个自然适合神经网络的应用领域是控制，它也是沿着智能控制[3]的方向演化。自治是控制系统设计者一个重大目标，而智能控制器是达到这个目标的一种方法。图 2 显示智能自治控制器的功能结构，这个智能自治控制器在涉及感觉的过程(设备)一端有一个界面，而在人和其他系统的一端有另一个界面(Antsaklis et al.,1996;Passino,1996)。系统有三个功能层，小结如下：

792

1. 执行层，它具有用于自适应控制和辨识的低层信号处理算法和控制算法。

2. 协调层，它通过监管诸如调谐、监督、危机管理和计划等事项提供执行层和管理层之间的联系。

3. 管理和组织层，它提供较低层的功能监督和对人的界面的管理。

既然经典控制是植根于线性微分方程组理论，智能控制主要是基于规则的，因为在其使用中涉及的相关性非常复杂以致不允许有解析的表示。为了处理这种相关性，使用模糊系统数学和神经网络是合适的。模糊系统[4]的功能在于它们的能力：(1)量化语言输入，(2)快速给出复杂的和通常未知的系统输入－输出规则的工作近似。神经网络的功能在于它们从数据中学习的能力。在神经网络和模糊系统之间存在一个自然的最佳协同，使得它们的混合对智能控制和其他应用而言是一个强有力的工具。

下面转入信号处理，它也是神经网络另一个有丰富应用的领域，这是因为神经网络的非线性和自适应特征(Haykin,1996)。对于在实际中遇到的信息承载信号(例如语音信号、雷达信号和声纳信号)，产生它们的大多数物理现象都是由非平稳和复杂的非线性动态系统控制，使得它们的精确数学描述成为不可能。为了在所有时间利用这种信号的所有信息内容，我们需要用于信号处理的智能机器[5]。它的设计解决下列关键论题：

- 非线性性，它使得提取输入信号的高阶统计成为可能。

- 学习和自适应，利用它们系统可以学习自身内嵌的环境的固有物理机制和在连续基础上自适应环境的缓慢变化。
- 注意机制，凭借它系统通过和最终用户交互或者以自组织的方式，能够集中它的计算能力针对图像的某一特别的点或空间中的特定位置，进行更详细的分析[6]

图3表示用于信号处理的智能机器的功能结构，它涉及操作的3个层次：

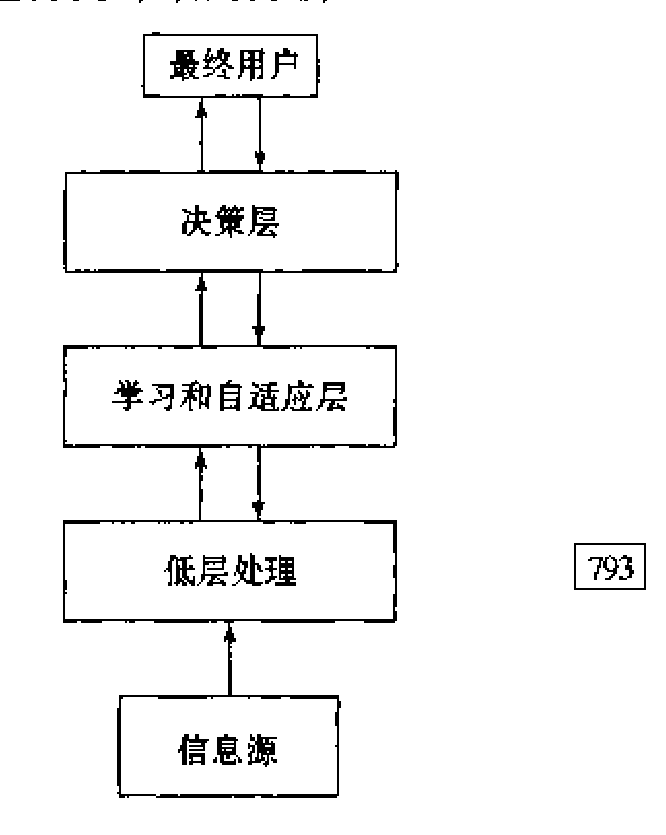


图3　用于信号处理的智能机器功能结构

1. *低层处理*，它的目的是对收到的信号作预处理，为第二层作准备。预处理涉及利用滤波削减噪声效果和其他高级信号处理操作，如时频分析[7]。时频分析的目标是描述信号的谱内容如何演变以及理解一个时变谱是什么。具体地，把收到信号的一维(时间)表示变换为二维图像，一维代表时间而另一维代表频率。时频分析提供一个有效方法，用于以一种远比原始时域形式清楚的方式突出收到信号的非平稳特性。

793

2. *学习和自适应层*，其中记忆(长期的和短期的)和注意机制被嵌入系统设计中。例如，用系统所处环境的足够大的数据集使多层感知器经历监督学习，环境的整体统计信息被储存在网络的突触权值中。为了考虑环境随时间的缓慢统计变化，一个盲自适应系统(即在无监督方式下运行的连续学习子系统)附加在多层感知器的输出端。学习过程也包括提供一个注意网络[8]，凭借它系统可以集中它的注意于收到信号的重要特征，这可以在需要时通过"选通"(gating)从较低层到较高层之间的信息流实现。

3. *决策层*，其中系统作出最终判决。判决可以是感兴趣的目标是否出现在收到的诸如雷达或声纳的信号中，或者在数字通信中收到的信息比特是否对应符号1或0；在决策中也提供置信级。

我们并不主张这里描述的系统是在系统中智能可以嵌入模式分类、控制和信号处理的惟一方式。相反，它们代表能实现这个重要目标的系统化方法。尽管它们存在应用领域的差异，它们确实具有一些共同特征(Valvanis and Saridis, 1992; Passino, 1996)：

- 从较低层到较高层和相反方向，存在双向信息流。
- 较高层经常关心系统的那些处理时间较慢、范围较广和横向时间较长的行为。
- 当我们从较低层移到较高层时随着精度的降低智能在升高。
- 在较高层，粒度有所下降(即模型的抽象性上升)。

我们在第1章通过将人脑描述为巨大的信息处理机器开始(人工)神经网络的讨论，人脑是神经网络的激励源泉。以智能机器的简短说明结束本书是合适的，智能机器是用人工手段进行信息处理的最高级。建立智能机器的努力将继续下去。

794

注释和参考文献

[1]　从不同角度对智能进行的原理性讨论，参看 Ackerman(1990)，Albus(1991)和 Kosko(1992)。

[2]　Viterbi 算法最初由 Viterbi 发展用于解决通信理论中的卷积解码问题。关于 Viterbi 算法

的指导性处理，参见 Forney(1973)。

关于模式分类应用中涉及卷积网络(在第 4 章描述)和 Viterbi 算法的联合使用，参见 LeCun et al.（1997,1998)。

[3] 智能控制在 White and Sofge(1992)，Antsaklis and Passino(1993)，Gupta and Sinha(1996)和 Tzefestas(1997)等编辑的书籍中讨论。

[4] 模糊理论由 Zadeh(1965,1973)创立，为处理语言变量(即用自然语言描述的概念)提供数学工具。以书本形式处理模糊逻辑，参看 Dubois and Prade(1980)。在 Kosko(1997)的书中，采用一种不同的观点：模糊系统被看作函数逼近器。其中证明模糊系统能模拟任何连续函数或者系统，只要模糊系统使用足够多的规则。

[5] 电气和电子工程师学会(Institute of Electrical and Electronic Engineers, IEEE)会刊 1998 年的一期专刊讨论智能信号处理的主题(Haykin and Kosko,1998)。

[6] 用于分层集中或选择注意的自组织系统在 Fukushima(1988a)中描述。系统是由 Fukushima(1975,1988b)创立的分层神经认知机的变形。系统能够在具有多个字符的图像中集中注意于单个字符或者集中注意于变形很大且被噪声损害的字符。

自组织注意机制也具有由 Carpenter and Grossberg(1987,1995)开创的自适应谐振理论(adaptive resonance theory, ART)的特征。用于自适应模式识别的 ART 涉及自底向上的滤波和自顶向下的模板匹配的组合。

795

[7] 建立在经典 Fourier 理论上的时频分析的许多方面的细节处理，参看 Cohen(1995)的书籍。

Wigner 分布为双线性/二次时频表示的重要工具，关于 Wigner 分布的理论和应用，参看 Mecklenbräuker and Hlawatsch(1997)的书籍。

对于用尺度而不用频率思考的另一种角度，参见 Vetterli and Koračević(1995)关于小波(wavelet)和子带编码的相关论题的书籍。

[8] 在 van de Laar et al.(1997)中描述用于选择性转换视觉注意的神经网络模型。这个模型根据所完成的任务通过调制在预注意阶段的信息流能够学会集中它的注意于重要特征。

参考文献

Aarts, E., and J. Korst, 1989. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing,* New York: Wiley.

Abarbanel, H.D.I., 1996. *Analysis of Observed Chaotic Data,* New York: Springer-Verlag.

Abraham, R.H., and C.D. Shaw, 1992. *Dynamics of the Geometry of Behavior,* Reading, MA: Addison-Wesley.

Abu-Mostafa, Y.S., 1995. "Hints," *Neural computation,* vol. 7, pp. 639–671.

Abu-Mostafa, Y.S., 1990. "Learning from hints in Neural Networks," *Journal of Complexity,* vol. 6, pp. 192–198.

Abu-Mostafa, Y.S., 1989. "The Vapnik-Chervonenkis Dimension: Information Versus Complexity in Learning," *Neural Computation,* vol. 1, pp. 312–317.

Abu-Mostafa, Y.S., and J.M. St. Jacques, 1985. "Information capacity of the Hopfield model," *IEEE Transactions on Information Theory,* vol. IT-31, pp. 461–464.

Ackerman, P.L., 1990. "Intelligence." In S.C. Shapiro, ed., *Encyclopedia of Artificial Intelligence,* pp. 431–440, New York: Wiley (Interscience).

Ackley, D.H., G.E. Hinton, and T.J. Sejnowski, 1985. "A Learning Algorithm for Boltzmann Machines," *Cognitive Science,* vol. 9, pp. 147–169.

Aiyer, S.V.B., N. Niranjan, and F. Fallside, 1990. "A theoretical investigation into the performance of the Hopfield model," *IEEE Transactions on Neural Networks,* vol. 15, pp. 204–215.

Aizerman, M.A., E.M. Braverman, and L.I. Rozonoer, 1964a. "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control,* vol. 25, pp. 821–837.

Aizerman, M.A., E.M. Braverman, and L.I. Rozonoer, 1964b. "The probability problem of pattern recognition learning and the method of potential functions," *Automation and Remote Control,* vol. 25, pp. 1175–1193.

Akaike, H., 1974. "A new look at the statistical model identification," *IEEE Transactions on Automatic Control* vol. AC-19, pp. 716–723.

Akaike, H., 1970. "Statistical predictor identification," *Annals of the Institute of Statistical Mathematics,* vol. 22, pp. 202–217.

Albus, J.S., 1991. "Outline for a theory of intelligence," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 21, pp. 473–509.

Aleksander, I., and H. Morton, 1990. *An Introduction to Neural Computing,* London: Chapman and Hall.

Allport A., 1989. "Visual attention," In *Foundations of Cognitive Science,* M.I. Posner, ed., pp. 631–682, Cambridge, MA: MIT Press.

Al-Mashoug, K.A., and I.S. Reed, 1991. "Including hints in training neural nets," *Neural Computation,* vol. 3, pp. 418–427.

Alspector, J., R.B. Allen, A. Jayakumar, T. Zeppenfeld, and R. Meir, 1991. "Relaxation networks for large supervised learning problems," *Advances in Neural Information Processing Systems,* vol. 3, pp. 1015–1021, San Mateo, CA: Morgan Kaufmann.

Alspector, J., A. Jayakumar, and S. Luna, 1992. "Experimental evaluation of learning in a neural microsystem," *Advances in Neural Information Processing Systems,* vol. 4, pp. 871–878, San Mateo, CA: Morgan Kaufmann.

Alspector, J., R. Meir, B. Yuhas, A. Jayakumar, and D. Lippe, 1993. "A parallel gradient descent method for learning in analog VLSI neural networks," *Advances in Neural Information Processing Systems,* vol. 5, pp. 836–844, San Mateo, CA: Morgan Kaufmann.

Amari, S., 1998. "Natural gradient works efficiently in learning," *Neural Computation,* vol. 10, pp. 251–276.

Amari, S., 1997. Private communication.

Amari, S., 1993. "A universal theorem on learning curves," *Neural Networks,* vol. 6, pp. 161–166.

Amari, S., 1990. "Mathematical foundations of neurocomputing," *Proceedings of the IEEE,* vol. 78, pp. 1443–1463.

Amari, S., 1987. "Differential geometry of a parametric family of invertible systems—Riemanian metric, dual affine connections and divergence," *Mathematical Systems Theory,* vol. 20, pp. 53–82.

Amari, S., 1985. *Differential-Geometrical Methods in Statistics,* New York: Springer-Verlag.

Amari, S., 1983. "Field theory of self-organizing neural nets," *IEEE Transactions on Systems, Man, and Cybernetics* vol. SMC-13, pp. 741–748.

Amari, S., 1980. "Topographic organization of nerve fields," *Bulletin of Mathematical Biology,* vol. 42, pp. 339–364.

Amari, S., 1977a. "Neural theory of association and concept-formation," *Biological Cybernetics,* vol. 26, pp. 175–185.

Amari, S., 1977b. "Dynamics of pattern formation in lateral-inhibition type neural fields," *Biological Cybernetics,* vol 27, pp. 77–87.

Amari, S., 1972. "Characteristics of random nets of analog neuron-like elements," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-2, pp. 643–657.

Amari, S., 1967. "A theory of adaptive pattern classifiers," *IEEE Trans. Electronic Computers,* vol. EC-16, pp. 299–307.

Amari, S., and M.A. Arbib, 1977. "Competition and cooperation in neural nets," in J. Metzler, ed., *Systems Neuroscience,* pp. 119–165, New York. Academic Press.

Amari, S., and J.-F. Cardoso, 1997. "Blind source separation—Semiparametric statistical approach," *IEEE Transactions on Signal Processing,* vol. 45, pp. 2692–2700.

Amari, S., T.-P. Chen, and A. Cichoki, 1997. "Stability analysis of learning algorithms for blind source separation," *Neural Networks,* vol. 10, pp. 1345–1351.

Amari, S., A. Cichoki, and H.H. Yang, 1996. "A new learning algorithm for blind signal separation." *Advances in Neural Information Processing Systems,* vol. 8, pp. 757–763, Cambridge, MA: MIT Press.

Amari, S., and K. Maginu, 1988. "Statistical neurodynamics of associative memory," *Neural Networks,* vol. 1, pp. 63–73.

Amari, S., K. Yoshida, and K.-I. Kanatani, 1977. "A mathematical foundation for statistical neurodynamics," *SIAM Journal of Applied Mathematics,* vol. 33, pp. 95–126.

Amari, S., N. Murata, K.-R. Müller, M. Finke, and H. Yang, 1996a. "Statistical theory of overtraining—Is cross-validation asymptotically effective?" *Advances in Neural Information Processing Systems,* vol. 8, pp. 176–182, Cambridge, MA: MIT Press.

Ambros-Ingerson, J., R. Granger, and G. Lynch, 1990. "Simulation of paleo-cortex performs hierarchical clustering," *Science,* vol. 247, pp. 1344–1348.

Amit, D.J., 1989. *Modeling Brain Function: The World of Attractor Neural Networks,* New York: Cambridge University Press.

Anastasio T.J., 1995. "Vestibulo-ocular reflex: Performance and plasticity," In M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* Cambridge, MA: MIT Press.

Anastasio, T.J., 1993. "Modeling vestibulo-ocular reflex dynamics: From classical analysis to neural networks," in F. Eeckman, ed., *Neural Systems: Analysis and Modeling,* pp. 407–430, Norwell, MA: Kluwer.

Anastasio, T.J., 1991. "A recurrent neural network model of velocity storage in the vestibulo-ocular reflex," *Advances in Neural Information Processing Systems,* vol. 3, pp. 32–38, San Mateo, CA: Morgan Kaufmann.

Anderson, J.A., 1995. *Introduction to Neural Networks,* Cambridge, MA: MIT Press.

Anderson, J.A., 1993. "The BSB model: A simple nonlinear autoassociative neural network," in *Associative Neural Memories* (M. Hassoun, ed.) pp. 77–103, Oxford: Oxford University Press.

Anderson, J.A., 1988. "General introduction," *Neurocomputing: Foundations of Research* (J.A. Anderson and E. Rosenfeld, eds.), pp. xiii–xxi, Cambridge, MA: MIT Press.

Anderson, J.A., 1983. "Cognitive and psychological computation with neural models," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-13, pp. 799–815.

Anderson, J.A., 1972. "A simple neural network generating an interactive memory," *Mathematical Biosciences,* vol. 14, pp. 197–220.

Anderson, J.A., and G.L. Murphy, 1986. "Concepts in connectionist models," in *Neural Networks for Computing,* J.S. Denker, ed., pp. 17–22, New York: American Institute of Physics.

Anderson, J.A., and E. Rosenfeld, eds., 1988. *Neurocomputing: Foundations of Research,* Cambridge, MA: MIT Press.

Anderson, J.A., A. Pellionisz, and E. Rosenfeld. eds., 1990a. *Neurocomputing 2: Directions for Research,* Cambridge, MA: MIT Press.

Anderson, J.A., J.W. Silverstein, S.A. Ritz, and R.S. Jones, 1977. "Distinctive features, categorical perception, and probability learning: Some applications of a neural model," *Psychological Review,* vol. 84, pp. 413–451.

Anderson, J.A., and J.P. Sutton, 1995. "A network of networks: Computation and neurobiology," *World Congress on Neural Networks,* vol. I, pp. 561–568.

Anderson, J.A., M.T. Gately, P.A. Penz. and D.R. Collins, 1990b. "Radar signal categorization using a neural network," *Proceedings of the IEEE,* vol. 78, pp. 1646–1657.

Anderson, T.W., 1984. *An Introduction to Multivariate Statistical Analysis,* 2nd edition, New York: Wiley.

Andreou, A.G., 1994. "On physical models of neural computation and their analog VLSI implementation," *Proceedings of the 1994 Workshop on Physics and Computation,* IEEE Computer Society Press, pp. 255–264, Los Alamitos, CA.

Andreou, A.G., K.A. Boahen, P.O. Pouliqueen, A. Pasavoic, R.E. Jenkins, and K. Strohbehn, 1991. "Current-

mode subthreshold MOS circuits for analog VLSI neural systems," *IEEE Transactions on Neural Networks*, vol. 2, pp. 205–213.

Andreou, A.G., R.C. Meitzler, K. Strohbehn, and K.A. Boahen, 1995. "Analog VLSI neuromorphic image acquisition and pre-processing systems." *Neural Networks*, vol. 8, pp. 1323–1347.

Andrews, R., and J. Diederich, eds., 1996. *Proceedings of the Rule Extraction from Trained Artificial Neural Networks Workshop*, University of Sussex, Brighton, UK.

Ansaklis, P.J., M. Lemmon, and J.A. Stiver, 1996. "Learning to be autonomous," In M.D. Gupta and N.K. Sinha, eds., *Intelligent Control Systems*, pp. 28–62. New York: IEEE Press.

Ansari, N., and E. Hou, 1997. *Computational Intelligence for Optimization*, Norwell, MA: Kluwer.

Anthony, M., and N. Biggs, 1992. *Computational Learning Theory*, Cambridge: Cambridge University Press.

Antsaklis, P.J., and K.M. Passino, eds., 1993. *An Introduction to Intelligent and Automatic Control*, Norwell, MA: Kluwer.

Arbib, M.A., 1989. *The Metaphorical Brain*, 2nd edition, New York: Wiley.

Arbib, M.A., 1987. *Brains, Machines, and Mathematics*, 2nd edition, New York: Springer-Verlag.

Arbib, M.A., ed. 1995. *The Handbook of Brain Theory and Neural Networks*, Cambridge, MA: MIT Press.

Arrowsmith, D.K., and C.M. Place, 1990. *An Introduction to Dynamical Systems*, Cambridge: Cambridge University Press.

Artola, A., and W. Singer, 1987. "Long-term potentiation and NMDA receptors in rat visual cortex," *Nature*, vol. 330, pp. 649–652.

Ash, R.E., 1965. *Information Theory*, New York: Wiley.

Ashby, W.R., 1960. *Design for a Brain*, 2nd edition, New York: Wiley.

Ashby, W.R., 1952. *Design for a Brain*, New York: Wiley.

Aspray, W., and A. Burks, 1986. *Papers of John von Neumann on Computing and Computer Theory*, Charles Babbage Institute Reprint Series for the History of Computing, vol. 12. Cambridge, MA: MIT Press.

Åström, K.J., and T.J. McAvoy, 1992. "Intelligent control: An overview and evaluation," In *Handbook of Intelligent Control*, D.A. White and D.A. Sofge, eds., New York: Van Nostrand Reinhold.

Atherton, D.P., 1981. *Stability of Nonlinear Systems*, Chichester, UK: Research Studies Press.

Atick, J.J., 1992. "Could information theory provide an ecological theory of sensory processing?" *Network: Computation in Neural Systems*, vol. 3, pp. 213–251.

Atick, J.J., and A.N. Redlich, 1992. "What does the retina know about natural scenes," *Neural Computation*, vol. 4, pp. 196–210.

Atick, J.J., and A.N. Redlich, 1990. "Towards a theory of early visual processing," *Neural Computation*, vol. 2, pp. 308–320.

Atick, J.J., P.A. Griffin, and A.N. Redlich, 1996. "Statistical approach to shape from shading: Reconstruction of three-dimensional face surfaces from single two-dimensional images," *Neural Computation*, vol. 8, pp. 1321–1340.

Atiya, A.F., 1987. "Learning on a general network," In *Neural Information Processing Systems*, D.Z. Anderson, ed., pp. 22–30, New York: American Institute of Physics.

Atiya, A.F., and Y.S. Abu-Mostafa, 1993, "An analog feedback associative memory," *IEEE Transactions on Neural Networks*, vol. 4, pp. 117–126.

Attneave, F., 1954. "Some informational aspects of visual perception," *Psychological Review*, vol. 61, pp. 183–193.

Back, A.D., and A.S. Weigend, 1998. "A first application of independent component analysis to extracting structure from stock returns," *International Journal of Neural Systems*, vol. 9, Special Issue on Data Mining in Finance, to appear.

Back, A.D., and A.C. Tsoi, 1991. "FIR and IIR synapses, a new neural network architecture for time series modeling," *Neural Computation*, vol. 3, pp. 375–385.

Back, A.D., and A.C. Tsoi, 1998. "A low-sensitivity recurrent neural network," *Neural Computation*, vol. 10, pp. 165–188.

Baldi, P., and K. Hornik, 1989. "Neural networks and principal component analysis: Learning from examples without local minimum," *Neural Networks*, vol. 1, pp. 53–58.

Bantine, W.L., and A.S. Weigend, 1994. "Computing second derivatives in feed-forward networks: A review," *IEEE Transactions on Neural Networks*, vol. 5, pp. 480–488.

Baras, J.S., and A. LaVigna, 1990. "Convergence of Kohonen's learning vector quantization," *International Joint Conference on Neural Networks*, vol. III, pp. 17–20, San Diego, CA.

Barlow, H.B., 1989. "Unsupervised learning," *Neural Computation*, vol. 1, pp. 295–311.

Barlow, H.B., 1985. "Cognitronics: methods for acquiring and holding cognitive knowledge," Unpublished manuscript.

Barlow, H.B., 1959. "Sensory mechanisms, the reduction of redundancy, and intelligence," in *The Mechanisation of Thought Processes, National Physical Laboratory Symposium No. 10*, Her Majesty's Stationary Office, London.

Barlow, H., and P. Földiák, 1989. "Adaptation and decorrelation in the cortex," in *The Computing Neuron*, R. Durbin, C. Miall, and G. Mitchison, eds., pp. 54–72. Reading, MA: Addison-Wesley.

Barnard, E., and D. Casasent, 1991. "Invariance and neural nets," *IEEE Transactions on Neural Networks*, vol. 2, pp. 498–508.

Barron, A.R., 1992. "Neural net approximation," in *Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 69–72, New Haven, CT.: Yale University.

Barron, A.R., 1993. "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information Theory*, vol. 39, pp. 930–945.

Bartlett, P.L., 1997. "For valid generalization, the size of the weights is more important than the size of the network," *Advances in Neural Information Processing Systems*, vol. 9, pp. 134–140, Cambridge, MA: MIT Press.

Barto, A.G., 1992. "Reinforcement learning and adaptive critic methods," in *Handbook of Intelligent Control*, D.A. White and D.A. Sofge, eds., pp. 469–491, New York: Van Nostrand Reinhold.

Barto, A.G., S.J. Bradtke, and S. Singh, 1995. "Learning to act using real-time dynamic programming," *Artificial Intelligence*, vol. 72, pp. 81–138.

Barto, A.G., R.S. Sutton, and C.W. Anderson, 1983. "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 834–846.

Basar, E., ed., 1990. *Chaos in Brain Function*, New York: Springer-Verlag.

Bashkirov, O.A., E.M. Braverman, and I.B. Muchnik, 1964. "Potential function algorithms for pattern recognition learning machines," *Automation and Remote Control*, vol. 25, pp. 629–631.

Battiti, R., 1992. "First- and second-order methods for learning: Between steepest descent and Newton's method," *Neural Computation*, vol. 4, pp. 141–166.

Bauer, H.-U., and K.R. Pawelzik, 1992. "Quantifying the neighborhood preservation of self-organizing feature maps," *IEEE Transactions on Neural Networks*, vol. 3, pp. 570–579.

Bauer, H.-U., R. Der, and M. Hermman, 1996. "Controlling the magnification factor of self-organizing feature maps," *Neural Computation*, vol. 8, pp. 757–771.

Baum, E.B., 1991. "Neural net algorithms that learn in polynomial time from examples and queries," *IEEE Transactions on Neural Networks*, vol. 2, pp. 5–19.

Baum, E.B., and D. Haussler, 1989. "What size net gives valid generalization?" *Neural Computation*, vol. 1, pp. 151–160.

Baum, E.B., and F. Wilczek, 1988. "Supervised learning of probability distributions by neural networks," in D.Z. Anderson, ed., pp. 52–61, New York: American Institute of Physics.

Beaufays, F., and E.A. Wan, 1994. "Relating real-time backpropagation and backpropagation-through-time: An application of flow graph interreciprocity," *Neural Computation*, vol. 6, pp. 296–306.

Becker, S., 1996. "Mutual information maximization: models of cortical self-organization," *Network: Computation in Neural Systems*, vol. 7, pp. 7–31.

Becker, S., 1993. "Learning to categorize objects using temporal coherence," *Advances in Neural Information Processing Systems*, vol. 5, pp. 361–368, San Mateo, CA: Morgan Kaufmann.

Becker, S., 1991. "Unsupervised learning procedures for neural networks," *International Journal of Neural Systems*, vol. 2, pp. 17–33.

Becker, S., and G.E. Hinton, 1993. "Learning mixture models of spatial coherence," *Neural Computation*, vol. 5, pp. 267–277.

Becker, S., and G.E. Hinton, 1992. "A self-organizing neural network that discovers surfaces in random-dot stereograms," *Nature (London)*, vol. 355, pp. 161–163.

Beckerman, M., 1997. *Adaptive Cooperative Systems*, New York: Wiley (Interscience).

Bell, A.J., and T.J. Sejnowski, 1995. "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 6, pp. 1129–1159.

Bellman, R., 1961. *Adaptive Control Processes: A Guided Tour*, Princeton, NJ: Princeton University Press.

Bellman, R., 1957. *Dynamic Programming*, Princeton, NJ: Princeton University Press.

Bellman, R., and S.E. Dreyfus, 1962. *Applied Dynamic Programming*, Princeton, NJ: Princeton University Press.

Bengio, Y., 1996. *Neural Networks for Speech and Sequence Recognition*. London: International Thomson Computer Press.

Bengio, Y., P. Simard, and P. Frasconi, 1994. "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, pp. 157–166.

Benveniste, A., M. Métivier, and P. Priouret, 1987. *Adaptive Algorithms and Stochastic Approximation*, New York: Springer-Verlag.

Bertero, M., T.A. Poggio, and V. Torre, 1988. "Ill-posed problems in early vision," *Proceedings of the IEEE*, vol. 76, pp. 869–889.

Bertsekas, D.P., 1995. *Dynamic Programming and Optimal Control*, vol. I and vol. II, Belmont, MA: Athenas Scientific.

Bertsekas, D.P., 1995. *Nonlinear Programming*, Belmont, MA: Athenas Scientific.

Bertsekas, D.P., and J.N. Tsitsiklis, 1996. *Neuro-Dynamic Programming,* Belmont, MA: Athenas Scientific.

Beymer, D., and T. Poggio, 1996. "Image representations for visual learning," *Science,* vol. 272, pp. 1905–1909.

Bienenstock, E.L., L.N. Cooper, and P.W. Munro, 1982. "Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex," *Journal of Neuroscience,* vol. 2, pp. 32–48.

Bishop, C.M., 1992. "Exact calculation of the Hessian matrix for the multi-layer perceptron," *Neural Computation,* vol. 4, pp. 494–501.

Bishop, C.M., 1995. *Neural Networks for Pattern Recognition.* Oxford: Clarendon Press.

Black, I.B., 1991. *Information in the Brain: A Molecular Perspective,* Cambridge, MA: MIT Press.

Blake, A., 1983. "The least-disturbance principle and week constraints," *Pattern Recognition Letters,* vol. 1, pp. 393–399.

Bliss, T.V.P., and T. Lomo, 1973. "Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforatant path," *J. Physiol,* vol. 232, pp. 331–356.

Blumer, A., A. Ehrenfeucht, D. Haussler, and M.K. Warmuth, 1989. "Learnability and the Vapnik–Chervonenkis Dimension," *Journal of the Association for Computing Machinery,* vol. 36, pp. 929–965.

Blumer, A., A. Ehrenfeucht, D. Haussler, and M.K. Warmuth, 1987. "Occam's razor," *Information Processing Letters,* vol. 24, pp. 377–380.

Boahen, K.A., 1996. "A retinomorphic vision system," *IEEE Micro,* vol. 16, no. 5, pp. 30–39.

Boahen, K.A., and A.G. Andreou, 1992. "A contrast sensitive silicon retina with reciprocal synapses," *Advances in Neural Information Processing Systems,* vol. 4, pp. 764–772, San Mateo, CA: Morgan Kaufmann.

Boahen, K.A., P.O. Pouliqueen, A.G. Andreou, and R.E. Jenkins, 1989. "A heterassociative memory using current-mode analog VLSI circuits," *IEEE Transactions on Circuits and Systems,* vol. CAS-36, pp. 747–755.

Bodenhausen, U., and A. Waibel, 1991. "The tempo 2 algorithm: Adjusting time-delays by supervised learning," *Advances in Neural Information Processing Systems,* vol. 3, pp. 155–161, San Mateo, CA: Morgan Kaufmann.

Boltzmann, L., 1872. "Weitere studien über das Wärmegleichgewicht unter gasmolekülen," *Sitzungsberichte der Mathematisch-Naturwissenschaftlichen Classe der Kaiserlichen Akademie der Wissenschaften,* vol. 66, pp. 275–370.

Boser, B., I. Guyon, and V.N. Vapnik, 1992. "A training algorithm for optimal margin classifiers," *Fifth Annual Workshop on Computational Learning Theory,* pp. 144–152. San Mateo, CA: Morgan Kaufmann.

Boser, B.E., E. Säckinger, J. Bromley, Y. LeCun, and L.D. Jackel, 1992. "Hardware requirements for neural network pattern classifiers," *IEEE Micro,* vol. 12, pp. 32–40.

Bourlard, H.A., and N. Morgan, 1994. *Connectionist Speech Recognition: A Hybrid Approach,* Boston: Kluwer.

Bourlard, H.A., and C.J. Wellekens, 1990. "Links between Markov models sand multilayer perceptrons," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-12, pp. 1167–1178.

Box, G.E.P., and G.M. Jenkins, 1976. *Time Series Analysis: Forecasting and Control,* San Francisco: Holden Day.

Braitenberg, V., 1967. "Is the cerebella cortex a biological clock in the millisecond range?" in *The Cerebellum. Progress in Brain Research,* C.A. Fox and R.S. Snider, eds., vol. 25 pp. 334–346, Amsterdam: Elsevier.

Braitenberg, V., 1990. "Reading the structure of brains," *Network: Computation in Neural Systems,* vol. 1, pp. 1 -12.

Braitenberg, V., 1986. "Two views of the cerebral cortex," in *Brain Theory,* G. Palm and A. Aertsen, eds., pp. 81–96. New York: Springer-Verlag.

Braitenberg, V., 1984. *Vehicles: Experiments in Synthetic Psychology,* Cambridge, MA: MIT Press.

Braitenberg, V., 1977. *On the Texture of Brains,* New York: Springer-Verlag.

Bregman, A.S., 1990. *Auditory Scene Analysis: The Perceptual Organization of Sound,* Cambridge, MA: MIT Press.

Breiman, L., 1996a. "Bagging predictors," *Machine Learning,* vol. 24, pp. 123–140.

Breiman, L., 1996b. "Bias, variance, and arcing classifiers," *Technical Report 460,* Statistics Department, University of California, Berkeley, Calif.

Breiman, L., J. Friedman, R. Olshen, and C. Stone, 1984. *Classification and Regression Trees,* New York: Chapman and Hall.

Bridle, J.S., 1990a. "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neuro-computing: Algorithms, Architectures and Applications,* F. Fougelman-Soulie and J. Hérault, eds., New York: Springer-Verlag.

Bridle, J.S., 1990b. "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters," *Advances in Neural Information Processing Systems,* vol. 2, pp. 211–217, San Mateo, CA: Morgan Kaufmann.

Brodal, A., 1981. *Neurological Anatomy in Relation to Clinical Medicine,* 3rd edition, New York: Oxford University Press.

Brodmann, K., 1909. *Vergleichende Lokalisationslehre der Grosshirnrinde,* Leipzig: J.A. Barth.

Brogan, W.L., 1985. *Modern Control Theory,* 2nd edition. Englewood Cliffs, NJ: Prentice-Hall.

Broomhead, D.S., and D. Lowe, 1988. "Multivariable functional interpolation and adaptive networks," *Complex Systems,* vol. 2, pp. 321–355.

Brown, T.H., E.W. Kairiss, and C.L. Keenan, 1990. "Hebbian synapses: Biophysical mechanisms and algorithms," *Annual Review of Neuroscience,* vol. 13, pp. 475–511.

Bruck, J., 1990. "On the convergence properties of the Hopfield model," *Proceedings of the IEEE,* vol. 78, pp. 1579–1585.

Bryson, A.E., Jr., and Y.C. Ho, 1969. *Applied Optimal Control,* Blaisdell (Revised printing, 1975, Hemisphere Publishing, Washington, DC).

Burg, J.P., 1975. *Modern Spectral Estimation,* Ph.D. Thesis, Stanford University, Stanford, Calif.

Burges, C.J.C., 1998. "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery,* to appear.

Cacoullos, T., 1966. "Estimation of a multivariate density," *Annals of the Institute of Statistical Mathematics (Tokyo),* vol. 18, pp. 179–189.

Caianiello, E.R., 1961. "Outline of a theory of thought-processes and thinking machines," *Journal of Theoretical Biology,* vol. 1, pp. 204–235.

Cameron, S.H., 1960. Tech. Report 60–600, *Proceedings of the Bionics Symposium,* pp. 197–212, Wright Air Development Division, Dayton, Ohio.

Cardoso, J.-F., 1998a. "Blind signal separation: A review," *Proceedings of the IEEE,* vol. 86, to appear.

Cardoso, J. F., 1998b. "Multidimensional independent component analysis," *Proceedings IEEE ICASSP,* Seattle, WA, May.

Cardoso, J.-F., 1997. "Infomax and maximum likelihood for blind source separation," *IEEE Signal Processing Letters,* vol. 4, pp. 112–114.

Cardoso, J.-F., 1996. "Entropic contrasts for source separation," Presented at the NIPS 96 *Workshop on Blind Signal Processing* organized by A. Cichoki at Snomass, Colo. To appear as a chapter in the book *Unsupervised Adaptive Filtering,* S. Haykin, ed., New York: Wiley.

Cardoso, J.-F., and B. Laheld, 1996. "Equivariant adaptive source separation," *IEEE Transactions on Signal Processing,* vol. 44, pp. 3017–3030.

Cardoso, J.-F., and A. Souloumia, 1993. "Blind beamforming for non-Gaussian signals," *IEE Proceedings (London), Part F,* vol. 140, pp. 362–370.

Carpenter, G.A., and S. Grossberg, 1987. "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision, Graphics, and Image Processing,* vol. 37, pp. 54–115.

Carpenter, G.A., M.A. Cohen, and S. Grossberg, 1987. Technical comments on "Computing with neural networks," *Science,* vol. 235, pp. 1226–1227.

Carpenter, G.A., and S. Grossberg, 1995. "Adaptive resonance theory (ART)," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* pp. 79–82. Cambridge, MA: MIT Press.

Casdagli, M., 1989. "Nonlinear prediction of chaotic time-series," *Physica,* vol. 35D, pp. 335–356.

Černy, V., 1985. "Thermodynamic approach to the travelling salesman problem," *Journal of Optimization Theory and Applications,* vol. 45, pp. 41–51.

Changeux, J.P., and A. Danchin, 1976. "Selective stabilization of developing synapses as a mechanism for the specification of neural networks," *Nature,* vol. 264, pp. 705–712.

Chatterjee, C., V.P. Roychowdhhury, and E.K.P. Chong, 1998. "On relative convergence properties of principal component algorithms," *IEEE Transactions on Neural Networks,* vol. 9, pp. 319–329.

Chen, H., and R.-W Liu, 1992. "Adaptive distributed orthogonalization processing for principal components analysis," *International Conference on Acoustics, Speech, and Signal Processing,* vol. 2, pp. 293–296, San Francisco.

Chen, S., 1995. "Nonlinear time series modelling and prediction using Gaussian RBF networks with enhanced clustering and RLS learning," *Electronics Letters,* vol. 31, No. 2, pp. 117–118.

Chen, S., S. Billings, and P. Grant, 1990. "Non-linear system identification using neural networks," *International Journal of Control,* vol. 51, pp. 1191–1214.

Chen, S., B. Mulgrew, and S. McLaughlin, 1992. "Adaptive Bayesian feedback equalizer based on a radial basis function network," *IEEE International Conference on Communications,* vol. 3, pp. 1267–1271, Chicago.

Cherkassky, V., and F. Mulier, 1995. "Self-organization as an iterative kernel smoothing process," *Neural Computation,* vol. 7, pp. 1165–1177.

Cherkassky, V., and F. Mulier, 1998. *Learning from Data: Concepts, Theory and Methods,* New York: Wiley.

Cherry, E.C., 1953. "Some experiments on the recognition of speech, with one and with two ears," *Journal of the Acoustical Society of America,* vol. 25, pp. 975–979.

Cherry, E.C., and W.K. Taylor, 1954. "Some further experiments upon the recognition of speech, with one and with two ears," *Journal of Acoustical Society of America,* vol. 26, pp. 554–559.

Chester, D.L., 1990. "Why two hidden layers are better than one," *International Joint Conference on Neural*

Networks, vol. I, pp. 265–268, Washington, D.C.

Chinrungrueng, C., and C.H. Séquin (1994). "Optimal adaptive k-means algorithm with dynamic adjustment of learning rate," *IEEE Transactions on Neural Networks*, vol. 6, pp. 157–169.

Choey, M., and A.S. Weigend, 1996. "Nonlinear trading models through Sharp ratio maximization," in A. Weigend, Y.S. Abu-Mostafa, and A.-P.N. Refenes, eds., *Decision Technologies for Financial Engineering*, pp. 3–22, Singapore: World Scientific.

Churchland, P.S., 1986. *Neurophilosophy: Toward a Unified Science of the Mind/Brain*, Cambridge, MA: MIT Press.

Churchland, P.S., and T.J. Sejnowski, 1992. *The Computational Brain*, Cambridge, MA: MIT Press.

Cichocki, A., and R. Unbehauen, 1996. "Robust neural networks with on-line learning for blind identification and blind separation of sources," *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 43, pp. 894–906.

Cichocki, A., R. Ubenhauen, and E. Rummert, 1994. "Robust learning algorithms for blind separation of signals," *Electronics Letters*, vol. 30, pp. 1386–1387.

Cichocki, A., and L. Moszcsynski, L., 1992. "New learning algorithm for blind separation of sources," *Electronics Letters*, vol. 28, pp. 1986–1987.

Cleeremans, A., D. Servan-Schreiber, and J.L. McClelland, 1989. "Finite state automata and simple recurrent networks," *Neural Computation*, vol. 1, pp. 372–381.

Cohen, L., 1995. *Time-Frequency Analysis*, Englewood Cliffs, NJ: Prentice-Hall.

Cohen, M.A., 1992a. "The synthesis of arbitrary stable dynamics in non-linear neural networks II: Feedback and universality," *International Joint Conference on Neural Networks*, vol. I, pp. 141–146, Baltimore.

Cohen, M.A., 1992b. "The construction of arbitrary stable dynamics in nonlinear neural networks," *Neural Networks*, vol. 5, pp. 83–103.

Cohen, M.A., and S. Grossberg, 1983. "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 815–826.

Cohn, D., and G. Tesauro, 1992. "How tight are the Vapnik–Chervonenkis bounds?" *Neural Computation*, vol. 4, pp. 249–269.

Comon, P., 1995. "Supervised classification, a probabilistic approach," *European Symposium on Artificial Neural Networks*, pp. 111–128, Brussels, Belgium.

Comon, P., 1994. "Independent component analysis: A new concept?" *Signal Processing*, vol. 36, pp. 287–314.

Comon, P., 1991. "Independent component analysis," *Proceedings of International Signal Processing Workshop on Higher-order Statistics*, pp. 111–120, Chamrousse, France.

Constantine-Paton, M., H.T. Cline, and E. Debski, 1990. "Patterned activity, synaptic convergence, and the NMDA receptor in developing visual pathways," *Annual Review of Neuroscience*, vol. 13, pp. 129–154.

Cook, A.S., 1971. "The complexity of theorem-proving procedures," *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, pp. 151–158, New York.

Cook, P.A., 1986. *Nonlinear Dynamical Systems*, London: Prentice-Hall International.

Cooper, L.N., 1973. "A possible organization of animal memory and learning," *Proceedings of the Nobel Symposium on Collective Properties of Physical Systems*, B. Lundquist and S. Lundquist, eds., pp. 252–264, New York: Academic Press.

Cormen, T.H., C.E. Leiserson, and R.R. Rivest, 1990. *Introduction to Algorithms*. Cambridge, MA: MIT Press.

Cortes, C., and V. Vapnik, 1995. "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297.

Cottrell, M., and J.C. Fort, 1986. "A stochastic model of retinotopy: A self organizing process," *Biological Cybernetics*, vol. 53, pp. 405–411.

Cottrell, M., J.C. Fort, and G. Pagès, 1997. "Theoretical aspects of the SOM algorithm," *Proceedings of the Workshop on Self-Organizing Maps*, Espoo, Finland.

Cottrell, G.W., and J. Metcalfe, 1991. "EMPATH: Face, emotion, and gender recognition using holons," *Advances in Neural Information Processing Systems*, vol. 3, pp. 564–571, San Mateo, CA: Morgan Kaufmann.

Cottrell, G.W., P. Munro, and D. Zipser, 1987. "Learning internal representations from grey-scale images: An example of extensional programming," *Proceedings of the 9th Annual Conference of the Cognitive science Society*, pp. 461–473.

Courant, R., and D. Hilbert, 1970. *Methods of Mathematical Physics*, vol. I and II, New York: Wiley Interscience.

Cover, T.M., 1968. "Capacity problems for linear machines," In L. Kanal, ed., *Pattern Recognition*, pp. 283–289, Washington, DC: Thompson Book Co.

Cover, T.M., 1965. "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, vol. EC-14, pp. 326–334.

Cover, T.M., and P.E. Hart, 1967. "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 21–27.

Cover, T.M., and J.A. Thomas, 1991. *Elements of Information Theory*, New York: Wiley.

Cowan, J.D., 1990. "Neural networks: The early days," *Advances in Neural Information Processing Systems*, vol. 2, pp. 828 842, San Mateo, CA: Morgan Kaufmann.

Cowan, J.D., 1968. "Statistical mechanics of nervous nets," in *Neural Networks*, E.R. Caianiello, ed., pp. 181–188, Berlin: Springer-Verlag.

Cowan, J.D. 1967. "A Mathematical Theory of Central Nervous Activity," Ph.D. Thesis, University of London.

Cowan, J.D. 1965. "The problem of organismic reliability," *Progress in Brain Research*, vol. 17, pp. 9–63.

Cowan, J.D., and M.H. Cohen, 1969. "The role of statistical mechanics in neurobiology," *Journal of the Physical Society of Japan*, vol. 26, pp. 51–53.

Cowan, J.D., and D.H. Sharp, 1988. "Neural nets," *Quarterly Reviews of Biophysics*, vol. 21, pp. 365–427.

Cragg, B.G., and H.N.V. Tamperley, 1955. "Memory: The analogy with ferromagnetic hysteresis," *Brain*, vol. 78, part II. pp. 304–316.

Cragg, B.G., and H.N.V. Tamperley, 1954. "The organization of neurons: A cooperative analogy," *EEG Clinical Neurophysiology*, vol. 6, pp. 85–92.

Craik, K.J.W., 1943. *The Nature of Explanation*, Cambridge: Cambridge University Press.

Craven, P., and G. Wahba, 1979. "Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation," *Numerische Mathematik.*, vol. 31, pp. 377–403.

Crick, F., 1989. "The recent excitement about neural networks," *Nature*, vol. 337, pp. 129–132.

Crites, R.H., and A.G. Barto, 1996. "Improving elevator performance using reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 8, pp. 1017–1023, Cambridge, MA: MIT Press.

Crutchfield, J.P., J.D. Farmer, N.H. Packard, and R.S. Shaw, 1986. "Chaos," *Scientific American*, vol. 255(6), pp. 38–49.

Cybenko, G., 1995. "Q-learning: A tutorial and extensions." Presented at *Mathematics of Artificial Neural Networks*, Oxford University, England, July 1995.

Cybenko, G., 1989. "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303–314.

Cybenko, G., 1988. "Approximation by superpositions of a sigmoidal function," Urbana, IL.: University of Illinois.

Darken, C., and J. Moody, 1992. "Towards faster stochastic gradient search," *Advances in Neural Information Processing Systems*, vol. 4, pp. 1009–1016, San Mateo, CA: Morgan Kaufmann.

Darmois, G., 1953. "Analyse generale des liaisons stochastiques," *Rev. Inst. Internat. Stat.*, vol. 21, pp. 2–8.

Dasarathy, B.V., ed., 1991. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, Los Alamitos, CA: IEEE Computer Society Press.

Daubechies, I., 1990. "The wavelet transform, time-frequency," *IEEE Transactions on Information Theory*, vol. IT-36, pp. 961–1005.

Daubechies, I., 1992. *Ten Lectures on Wavelets*, SIAM.

Davis, P.J., 1963. *Interpolation and Approximation*, New York: Blaisdell.

Dayan, P., and G.E. Hinton, 1996. "Varieties of Helmholtz machine," *Neural Networks*, vol. 9, pp. 1385–1403.

Dayan, P., G.E. Hinton, R.M. Neal, and R.S. Zemel, 1995. "The Helmholtz machine," *Neural Computation*, vol. 7, pp. 889–904.

Debnath, L., and P. Mikusiński, 1990. *Introduction to Hilbert Spaces with Applications*, New York: Academic Press.

Deco, G., W. Finnoff, and H.G. Zimmermann, 1995. "Unsupervised mutual information criteria for elimination of overtraining in supervised multilayer networks," *Neural Computation*, vol. 7, pp. 86–107.

Deco, G., and D. Obradovic, 1996. *An Information-Theoretic Approach to Neural Computing*, New York: Springer.

de Figueiredo, R.J.P., 1980. "Implications and applications of Kolmogorov's superposition theorem," *IEEE Transactions on Automatic Control*, vol. AC-25, pp. 1227–1230.

de Figueiredo, R.J.P., and G. Chen, 1993. *Nonlinear Feedback Control Systems*, New York: Academic Press.

DeMers, D., and G. Cottrell, 1993. "Non-linear dimensionality reduction," *Advances in Neural Information Processing Systems*, vol. 5, pp. 580–587. San Mateo, CA: Morgan Kaufmann.

Dempster, A.P., N.M. Laird, and D.B. Rubin. 1977. "Maximum likelihood from incomplete data via the EM algorithm," (with discussion), *Journal of the Royal Statistical Society.*, B, vol. 39, pp. 1–38.

Denardo, E.V., 1967. "Contraction mappings in the theory underlying dynamic programming," *SIAM*, Review, vol. 9, pp. 165–177.

DeSieno, D., 1988. "Adding a conscience to competitive learning," *IEEE International Conference on Neural Networks*, vol. I, pp. 117–124, San Diego, CA.

deSilva, C.J.S., and Y. Attikiouzel, 1992. "Hopfield networks as discrete dynamical systems," *International Joint Conference on Neural Networks*, vol. III, pp. 115–120, Baltimore.

deVries, B., and J.C. Principe, 1992. "The gamma model—A new neural model for temporal processing,"

Neural Networks, vol. 5, pp. 565–576.

Devroye. L., 1991. "Exponential inequalities in nonparametric estimation," in *Nonparametric Functional Estimation and Related Topics,* G. Roussas, ed., pp. 31–44. Boston: Kluwer.

Diamantaras, K.I., and S.Y. Kung, 1996. *Principal Component Neural Networks: Theory and Applications,* New York: Wiley.

Dohrmann, C.R., H.R. Busby, and D.M. Trujillo, 1988. "Smoothing noisy data using dynamic programming and generalized cross-validation," *Journal of Biomechanical Engineering,* vol. 110, pp. 37–41.

Domany, E., J.L. van Hemmen, and K. Schulten, eds., 1991. *Models of Neural Networks,* New York: Springer-Verlag.

Dony, R.D., and S. Haykin, 1997. "Image segmentation using a mixture of principal components representation," *IEE Proceedings (London), Image and Signal Processing,* vol. 144, pp. 73–80.

Dony, R.D., and S. Haykin, 1995. "Optimally adaptive transform coding," *IEEE Transactions on Image Processing,* vol. 4, pp. 1358–1370.

Dorny, C.N., 1975. *A Vector Space Approach to Models and Optimization,* New York: Wiley (Interscience).

Douglas, S.C., and S. Haykin, 1997. "On the relationship between blind deconvolution and blind source separation," *Thirty-First Asilomar Conference on Signals, Systems, and Computers,* Pacific Grove, California, November.

Doyle, J.C., K. Glover, P. Khargonekar, and B. Francis, 1989. "State-space solutions to standard H_2 and H_∞ control problems," *IEEE Transactions on Automatic Control,* vol. AC-34, pp. 831–847.

Drucker, H., C. Cortes, L.D. Jackel, and Y. LeCun, 1994. "Boosting and other ensemble methods." *Neural Computation,* vol. 6, pp. 1289–1301.

Drucker, H., R.E. Schapire, and P. Simard, 1993. "Improving performance in neural networks using a boosting algorithm," *Advances in Neural Information Processing Systems,* vol. 5, pp. 42–49, Cambridge, MA: MIT Press.

Dubois, D., and H. Prade, 1980. *Fuzzy Sets and Systems: Theory and Applications,* New York: Academic Press.

Duda, R.O., and P.E. Hart, 1973. *Pattern Classification and Scene Analysis,* New York: Wiley.

Dunford, N., and J.T. Schwartz, 1966. *Linear Operators, Part 1,* New York: Wiley.

Durbin, R., C. Miall, and G. Mitchison, eds., 1989. *The Computing Neuron,* Reading, MA: Addison-Wesley.

Durbin, R., and D.E. Rumelhart, 1989. "Product units: A computationally powerful and biologically plausible extension to backpropagation networks," *Neural Computation,* vol. 1, pp. 133–142.

Durbin, R., and D. Willshaw, 1987. "An analogue approach to the travelling salesman problem using an elastic net method," *Nature,* vol. 326, pp. 689–691.

Dyn, N., 1987. "Interpolation of scattered data by radial functions," in *Topics in Multivariate Approximation,* C.K. Chui, L.L. Schumaker, and F.I. Uteras, eds., pp. 47–61, Orlando, FL: Academic Press.

Edelman, G.M., 1987. *Neural Darwinism,* New York: Basil Books.

Edelman, G.M., 1973. "Antibody structure and molecular immunology," *Science,* vol. 180, pp. 830–840.

Eeckman, F.H., 1988. "The sigmoid nonlinearity in prepyriform cortex," *Neural Information Processing Systems,* pp. 242–248, New York: American Institute of Physics.

Eeckman, F.H., and W.J. Freeman, 1986. "The sigmoid nonlinearity in neural computation: An experimental approach," *Neural Networks for Computing,* J.S. Denker, ed., pp. 135–145, New York: American Institute of Physics.

Eggermont, J.J., 1990. *The Correlative Brain: Theory and Experiment in Neural Interaction,* New York: Springer-Verlag.

El Hihi, S., and Y. Bengio, 1996. "Hierarchical recurrent neural networks for long-term dependencies," *Advances in Neural Information Processing Systems,* vol. 8, pp. 493–499, MIT Press.

Elman, J.L., 1990. "Finding structure in time," *Cognitive Science,* vol. 14, pp. 179–211.

Elman, J.L., E.A. Bates, M.H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett, 1996. *Rethinking Innateness: A Connectionist Perspective on Development,* Cambridge, MA: MIT Press.

Erwin, E., K. Obermayer, and K. Schulten, 1995. "Models of orientation and ocular dominance columns in the visual cortex: A critical comparison," *Neural Computation,* vol. 7, pp. 425–468.

Erwin, E., K. Obermayer, and K. Schulten, 1992a. "I: Self-organizing maps: Stationary states, metastability and convergence rate," *Biological Cybernetics,* vol. 67, pp. 35–45.

Erwin, E., K. Obermayer, and K. Schulten, 1992b. "II: Self-organizing maps: Ordering, convergence properties and energy functions," *Biological Cybernetics,* vol. 67, pp. 47–55.

Faggin, F., 1991. "VLSI implementation of neural networks," Tutorial Notes, *International Joint Conference on Neural Networks,* Seattle.

Faggin, F., and C. Mead, 1990. "VLSI Implementation of Neural Networks," *An Introduction to Neural and Electronic Networks,* S.F. Zornetzer, J.L. Davis, and C. Lau, eds., pp. 275–292, New York: Academic Press.

Fahlman, S.E., and C. Lebiere, 1990. "The cascade-correlation learning architecture," *Advances in Neural Information Processing Systems,* vol. 2, pp. 524–532, San Mateo, CA: Morgan Kaufmann.

Farmer, J.D., and J. Sidorowich, 1987. "Predicting chaotic time series," *Physical Review Letters*, vol. 59, pp. 845–848.

Feldkamp, L.A., and G.V. Puskorius, 1997. "Adaptive behavior from fixed weight networks," *Information Sciences*, vol. 98, pp. 217–235.

Feldkamp, L.A., and G.V. Puskorius, 1998. "A signal processing framework based on dynamic neural networks with application to problems in adaptation, filtering and classification," *Proceedings of the IEEE*, vol. 86, to appear.

Feldkamp, L.A., G.V. Puskorius, and P.C. Moore, 1997. "Adaptation from fixed weight networks," *Information Sciences*, vol. 98, pp. 217–235.

Feldman, J.A., 1992. "Natural computation and artificial intelligence," Plenary Lecture presented at the *International Joint Conference on Neural Networks*, Baltimore.

Feller, W., 1968. *An Introduction to Probability Theory and its Applications*, vol. 1, 3rd edition, New York: John Wiley; 1st edition, 1950.

Fischler, M.A., and O. Firschein, 1987. *Intelligence: The Eye, The Brain, and The Computer*, Reading, MA: Addison-Wesley.

Fisher, R.A., 1925. "Theory of statistical estimation," *Proceedings of the Cambridge Philosophical Society*, vol. 22, pp. 700–725.

Fix, E., and J.L. Hodges, 1951. "Discriminatory analysis: Nonparametric discrimination: Consistency properties," *USAF School of Aviation Medicine*, Project 21-49-004, Report no. 4, pp. 261–279, Randolph Field, Texas.

Fletcher, R., 1987. *Practical Methods of Optimization*, 2nd edition, New York: Wiley.

Fodor, J.A., 1983. *Modularity of Mind*, Cambridge, MA: MIT Press.

Fodor, J.A., and Z.W. Pylyshyn, 1988. "Connectionism and cognitive architecture: a critical analysis," *Cognition*, vol. 28, pp. 3–72.

Földiak, P., 1989. "Adaptive network for optimal linear feature extractions," *IEEE International Joint Conference on Neural Networks*, vol. I, pp. 401–405, Washington, DC.

Forcada, M.L., and R.C. Carrasco, 1995. "Learning the initial state of a second-order recurrent neural network during regular-language inference," *Neural Computation*, vol. 7, pp. 923–930.

Forney, G.D., Jr., 1973. "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–278.

Forte, J.C., and G. Pagés, 1995. "On the a.s. convergence of the Kohonen algorithm with a general neighborhood function," *Annals of Applied Probability*, vol. 5, pp. 1177–1216.

Forte, J.C., and G. Pagés, 1996. "Convergence of stochastic algorithm: From the Kushner and Clark theorem to the Lyapunov functional," *Advances in Applied Probability*, vol. 28, pp. 1072–1094.

Frasconi, P., M. Gori, and G. Soda, 1992. "Local feedback multilayered networks," *Neural Computation*, vol. 4, pp. 120–130.

Frasconi, P., and M. Gori, 1996. "Computational capabilities of local-feedback recurrent networks acting as finite-state machines," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1521–1524.

Fraser, A.M., 1989. "Information and entropy in strange attractors," *IEEE Transactions on Information Theory*, vol. 35, pp. 245–262.

Freeman, J.A., and D.M. Sakpura, 1991. *Neural Networks: Algorithms, Applications, and Programming Techniques*, Reading, MA: Addison-Wesley.

Freeman, W.J., 1995. *Societies of Brains*. Hillsdale, NJ: Lawrence Erlbaum.

Freeman, W.J., 1992. "Tutorial on neurobiology: From single neurons to brain chaos," *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, vol. 2, pp. 451–482.

Freeman, W.J., 1991. "The physiology of perception," *Scientific American*, vol. 264 (2), pp. 78–85.

Freeman, W.J., 1988. "Why neural networks don't yet fly: Inquiry into the neurodynamics of biological intelligence," *IEEE International Conference on Neural Networks*, vol. II, pp. 1–7, San Diego, CA.

Freeman, W.J., 1987. "Simulation of chaotic EEG patterns with a dynamic model of the olfactory system," *Biological Cybernetics*, vol. 56, pp. 139–150.

Freeman, W.J., 1975. *Mass Action in the Nervous System*, New York: Academic Press.

Frégnac, Y., and D. Schulz, 1994. "Models of synaptic plasticity and cellular analogs of learning in the developing and adult vertebrate visual cortex," *Advances in Neural and Behavioral Development*, vol. 4, pp. 149–235, Norwood, NJ: Neural Ablex.

Freund, Y., 1995. "Boosting a week learning algorithm by majority," *Information Computation*, vol. 121, pp. 256–285.

Freund, Y., and R.E. Schapire, 1997. "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of Computer and System Sciences*, vol. 55, pp. 119–139.

Freund, Y., and R.E. Schapire, 1996a. "Experiments with a new boosting algorithm," *Machine Learning: Proceedings of the Thirteenth International Conference*, pp. 148–156, Bari, Italy.

Freund, Y., and R.E. Schapire, 1996b. "Game theory, On-line prediction and boosting," *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pp. 325–332, Desenzano del Garda, Italy.

Friedman, J.H., 1995. "An overview of prediction learning and function approximation," In V. Cherkassky, J.H. Friedman, and H. Wechsler, eds., *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*, New York: Springer-Verlag.

Fukunaga, K., 1990. *Statistical Pattern Recognition*, 2nd edition, New York: Academic Press.

Fukushima, K., 1995. "Neocognitron: A model for visual pattern recognition," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, Cambridge, MA: MIT Press.

Fukushima, K., 1988a. "A hierarchical neural network model for selective attention," in *Neural Computers*, R. Eckmiller and C. von der Malsburg, eds., pp. 81–90, NATO ASI Series, New York: Springer-Verlag.

Fukushima, K., 1988b. "Neocognitron: A hierarchical neural network capable of visual pattern recognition." *Neural Networks*, vol. 1, pp. 119–130.

Fukushima, K., 1980. "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, 193–202.

Fukushima, K., 1975. "Cognitron: A self-organizing multi-layered neural network," *Biological Cybernetics*, vol. 20, pp. 121–136.

Fukushima, K., S. Miyake, and T. Ito, 1983. "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 826–834.

Funahashi, K., 1989. "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, pp. 183–192.

Gabor, D., 1954. "Communication theory and cybernetics," *IRE Transactions on Circuit Theory*, vol. CT-1, pp. 19–31.

Gabor, D., W.P.L. Wilby, and R. Woodcock, 1960. "A universal non-linear filter, predictor, and simulator which optimizes itself by a learning process," *Proceedings of the Institution of Electrical Engineers*, London, vol. 108, pp. 422–435.

Galland, C.C., 1993. "The limitations of deterministic Boltzmann machine learning," *Network*, vol. 4, pp. 355–379.

Gallant, A.R., and H. White, 1988. "There exists a neural network that does not make avoidable mistakes," *IEEE International Conference on Neural Networks*, vol. I, pp. 657–664, San Diego, CA.

Gallant, A.R., and H. White, 1992. "On learning the derivatives of an unknown mapping with multilayer feedforward networks," *Neural Networks*, vol. 5, pp. 129–138.

Gallant, S.I., 1993. *Neural Network Learning and Expert Systems*, Cambridge, MA: MIT Press.

Gallistel, C.R., 1990. *The Organization of Learning*, Cambridge, MA: MIT Press.

Gardner, E., 1987. "Maximum storage capacity in neural networks," *Electrophysics Letters*, vol. 4, pp. 481–485.

Garey, M.R., and D.S. Johnson, 1979. *Computers and Intractability*, New York: W.H. Freeman.

Gee, A.H., 1993. "Problem solving with optimization networks," Ph.D. dissertation, University of Cambridge.

Gee, A.H., S.V.B. Aiyer, and R. Prager, 1993. "An analytical framework for optimizing neural networks," *Neural Networks*, vol. 6, pp. 79–97.

Geisser, S., 1975. "The predictive sample reuse method with applications," *Journal of the American Statistical Association*, vol. 70, pp. 320–328.

Gelfand, A.E., and A.F.M. Smith, 1990. "Sampling-based approaches to calculating marginal densities," *Journal of the American Statistical Association*, vol. 85, pp. 398–409.

Geman, S., and D. Geman, 1984. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721–741.

Geman, S., E. Bienenstock, and R. Doursat, 1992. "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, pp. 1–58.

Gersho, A., 1982. "On the structure of vector quantizers," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 157–166.

Gersho, A., and R.M. Gray, 1992. *Vector Quantization and Signal Compression*, Norwell, MA: Kluwer.

Gerstein, G.L., P. Bedenbaugh, and A.M.H.J. Aersten, 1989. "Neural assemblies," *IEEE Transactions on Biomedical Engineering*, vol. 36, pp. 4–14.

Gibbs, J.W., 1902. "Elementary principles in statistical mechanics," reproduced in vol. 2 of *Collected Works of J. Willard Gibbs in Two Volumes*, New York: Longmans, Green and Co., 1928.

Gibson, G.J., and C.F.N. Cowan, 1990. "On the decision regions of multilayer perceptrons," *Proceedings of the IEEE*, vol. 78, pp. 1590–1599.

Gidas, B., 1985. "Global optimization via the Langevin equation," *Proceedings of 24th Conference on Decision and Control*, pp. 774–778, Ft. Lauderdale, FL.

Giles, C.L., 1996. "Dynamically driven recurrent neural networks: Models, learning algorithms, and applications," Tutorial #4, *International Conference on Neural Networks*, Washington, DC.

Giles, C.L., D. Chen, G.Z. Sun, H.H. Chen, Y.C. Lee, and M.W. Goudreau, 1995. "Constructive learning of recurrent neural networks: Limitations of recurrent cascade correlation with a simple solution," *IEEE Transactions on Neural Networks*, vol. 6, pp. 829–836.

Giles, C.L., T. Lin, and B.G. Horne, 1997. "Remembering the past: The role of embedded memory in recurrent neural network architectures," *Neural Networks for Signal Processing*, VII, Proceedings of the 1997 IEEE Workshop, IEEE Press, p. 34.

Giles, C.L., and T. Maxwell, 1987. "Learning, invariance, and generalization in higher-order neural networks," *Applied Optics*, vol. 26, pp. 4972–4978.

Giles, C.L., and B.G. Horne, 1994. "Representation of learning in recurrent neural network architectures," *Proceedings of the Eighth Yale Workshop on Adaptive and Learning Systems*, pp. 128–134, Yale University, New Haven, Ct.

Giles, C.L., C.B. Miller, D. Chen, H.H. Chen, G.Z. Sun, and Y.C. Lee, 1992. "Learning and extracting finite state automata with second-order recurrent neural networks," *Neural Computation*, vol. 4, pp. 393–405.

Giles, C.L., G.Z. Sun, H.H. Chen, Y.C. Lee, and D. Chen, 1990. "Higher order recurrent networks and grammatical inference," *Advances in Neural Information Processing Systems*, vol. 2, pp. 380–387, San Mateo, CA: Morgan Kaufmann.

Gill, P., S. Hammarling, W. Murray, M. Saunders, and M. Wright, 1986. "User's guide for LSSOL," Technical Report 86–1, Systems Optimization Laboratory, Stanford University, Stanford, CA.

Gill, P., and W. Murray, 1991. "Inertia-controlling methods for general quadratic programming," *SIAM Review*, vol. 33, pp. 1–36.

Girosi, F., and G. Anzellotti, 1992. "Rates of convergence of approximation by translates," *A.I. Memo 1288*, Artificial Intelligence Laboratory, MIT Cambridge, MA:

Girosi, F., M. Jones, and T. Poggio, 1995. "Regularization theory and neural networks architectures," *Neural Computation*, vol. 7, pp. 219–269.

Girosi, F., and T. Poggio, 1990. "Networks and the best approximation property," *Biological Cybernetics*, vol. 63, pp. 169–176.

Glauber, R.J., 1963. "Time-dependent statistics of the Ising model," *Journal of Mathematical Physics*, vol. 4, pp. 294–307.

Goggin, S.D.D., K.M. Johnson, and K. Gustafson, 1989. "Primary and recency effects due to momentum in back-propagation learning," *OCS Technical Report 89-25*, Boulder, CO.: University of Colorado.

Golden, R.M., 1996. *Mathematical Methods for Neural Network Analysis and Design*, Cambridge, MA: MIT Press.

Golden, R.M., 1986. "The 'Brain-State-in-a-Box' neural model is a gradient descent algorithm," *Journal of Mathematical Psychology*, vol. 30, pp. 73–80.

Goles, E., and S. Martínez, 1990. *Neural and Automata Networks*, Dondrecht, The Netherlands: Kluwer.

Golub, G.H., and C.G. Van Loan, 1996. *Matrix Computations*, 3rd edition, Baltimore: Johns Hopkins University Press.

Goodman, R.M., C.M. Higgins, J.W. Miller, and P. Smyth, 1992. "Rule-based neural networks for classification and probability estimation," *Neural Computation*, vol. 4, pp. 781–804.

Gori, M., and A. Tesi, 1992. "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 76–86.

Gorin, A., 1992. "Network structure, generalization and adaptive language acquisition," *Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 155–160, Yale University, New Haven, CT.

Goudreau, M.W., and C.L. Giles, 1995. "Using recurrent neural networks to learn the structure of interconnection networks," *Neural Networks*, vol. 8, pp. 793–804.

Goudreau, M.W., C.L. Giles, S.T. Chakradhar, and D. Chen, 1994. "First-order vs. second-order single-layer recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 5, pp. 511–513.

Granger, R., J. Whitson, J. Larson, and G. Lynch, 1994. "Non-Hebbian properties of LTP enable high-capacity encoding of temporal sequences," *Proceedings of the National Academy of Sciences of the U.S.A.*, to appear.

Grassberger, I., and I. Procaccia, 1983. "Measuring the strangeness of strange attractors," *Physica D*, vol. 9, pp. 189–208.

Graubard, S.R., ed., 1988. *The Artificial Intelligence Debate: False Starts, Real Foundations*, Cambridge, MA: MIT Press.

Gray, R.M., 1990. *Entropy and Information Theory*, New York: Springer-Verlag.

Gray, R.M., 1988. *Probability, Random Processes, and Ergodic Properties*, New York: Springer-Verlag.

Gray, R.M., 1984. "Vector quantization," *IEEE ASSP Magazine*, vol. 1, pp. 4–29.

Gray, R.M., and L.D. Davisson, 1986. *Random Processes: A Mathematical Approach for Engineers*, Englewood Cliffs, NJ: Prentice-Hall.

Green, M., and D.J.N. Limebeer, 1995. *Linear Robust Control*, Englewood Cliffs, NJ: Prentice-Hall.

Greenberg, H.J., 1988. "Equilibria of the brain-state-in-a-box (BSB) neural model," *Neural Networks*, vol. 1, pp. 323–324.

Gregory, R.L., 1970. *The Intelligent Eye*, Wiedefeld and Nicholson, London.

Grenander, U., 1983. *Tutorial in Pattern Theory*, Brown University, Providence, R.I.

Grewal, M.S., and A.P. Andrews, 1993. *Kalman Filtering: Theory and Practice*, Englewood Cliffs, NJ: Prentice-Hall.

Griffiths, L.J., and C.W. Jim, 1982. "An alternative approach to linearly constrained optimum beamforming," *IEEE Transactions on Antennas and Propagation*, vol. AP-30, pp. 27–34.

Grossberg, S., 1990. "Content-addressable memory storage by neural networks: A general model and global Liapunov method," In *Computational Neuroscience*, E.L. Schwartz, ed., pp. 56–65, Cambridge, MA: MIT Press.

Grossberg, S., 1988a. "Competitive learning: From interactive activation to adaptive resonance," in *Neural Networks and Natural Intelligence*, S. Grossberg, ed., Cambridge, MA: MIT Press.

Grossberg, S., 1988b. *Neural Networks and Natural Intelligence*, Cambridge, MA: MIT Press.

Grossberg, S., 1988c. "Nonlinear neural networks: Principles, mechanisms, and architectures," *Neural Networks*, vol. 1, pp. 17–61.

Grossberg, S., 1982. *Studies of Mind and Brain*, Boston: Reidel.

Grossberg, S., 1980. "How does a brain build a cognitive code?" *Psychological Review*, vol. 87, pp. 1–51.

Grossberg, S., 1978a. "Decision, patterns, and oscillations in the dynamics of competitive systems with application to Volterra-Lotka systems," *J. Theoretical Biology*, vol. 73, pp. 101–130.

Grossberg, S., 1978b. "Competition, decision, and consensus," *J. Mathematical Analysis and Applications*, vol. 66, pp. 470–493.

Grossberg, S., 1977. "Pattern formation by the global limits of a nonlinear competitive interaction in n dimensions," *J. Mathematical Biology*, vol. 4, pp. 237–256.

Grossberg, S., 1976a. "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural detectors," *Biological Cybernetics*, vol. 23, pp. 121–134;

Grossberg, S., 1976b. "Adaptive pattern classification and universal recoding: II. Feedback, expectation, olfaction, illusions," *Biological Cybernetics*, vol. 23, pp. 187–202.

Grossberg, S., 1972. "Neural expectation: Cerebellar and retinal analogs of cells fired by learnable or unlearned pattern classes," *Kybernetik*, vol. 10, pp. 49–57.

Grossberg, S., 1969a. "A prediction theory for some nonlinear functional-difference equations," *Journal of Mathematical Analysis and Applications*, vol. 22, pp. 490–522.

Grossberg, S., 1969b. "On learning and energy-entropy dependence in recurrent and nonrecurrent signed networks," *Journal of Statistical Physics*, vol. 1, pp. 319–350.

Grossberg, S., 1968. "A prediction theory for some nonlinear functional-difference equations," *Journal of Mathematical Analysis and Applications*, vol. 21, pp. 643–694, vol. 22, pp. 490–522.

Grossberg, S., 1967. "Nonlinear difference—differential equations in prediction and learning theory," *Proceedings of the National Academy of Sciences*, USA, vol. 58, pp. 1329–1334.

Gupta, M.M., and N.K. Sinha, eds. 1996. *Intelligent Control Systems: Theory and Applications*, New York: IEEE Press.

Guyon, I., 1990. *Neural Networks and Applications*, Computer Physics Reports, Amsterdam: Elsevier.

Haffner, P., 1994. "A new probabilistic framework for connectionist time alignment," *Proceedings of ICSLP 94*, pp. 1559–1562, Yokohama, Japan.

Haffner, P., M. Franzini, and A. Waibel, 1991. "Integrating time alignment and neural networks for high performance continuous speech recognition," *Proceedings of IEEE ICASSP 91*, pp. 105–108.

Haft, M., and J.L. van Hemmen, 1998. "Theory and implementations of infomax filters for the retina," *Network: Computations in Neural Systems*, vol. 9, pp. 39–71.

Hagiwara, M., 1992. "Theoretical derivation of momentum term in back-propagation," *International Joint Conference on Neural Networks*, vol. I, pp. 682–686, Baltimore.

Hajek, B., 1985. "A tutorial survey of theory and applications of simulated annealing," *Proceedings of the 24th Conference on Decision and Control*, IEEE Press, pp. 755–760, Ft. Lauderdale, Fla.

Hajek, B., 1988. "Cooling schedules for optimal annealing," *Mathematics of Operations Research*, vol. 13, pp. 311–329.

Hammerstrom, D., 1993a. "Neural networks at work," *IEEE Spectrum*, vol. 30, no. 6, pp. 26–32.

Hammerstrom, D., 1993b. "Working with neural networks," *IEEE Spectrum*, vol. 30, no. 7, pp. 46–53.

Hammerstrom, D., and S. Rahfuss, 1992. "Neurocomputing hardware: Present and future," *Swedish National Conference on Connectionism*, Skovade, Sweden, September.

Hampshire, J.B., and B. Pearlmutter, 1990. "Equivalence proofs for multilayer perceptron classifiers and Bayesian discriminant function," *Proceedings of the 1990 Connectionist Models Summer School*, pp. 159–172, San Mateo, CA: Morgan Kaufmann.

Hampson, S.E., 1990. *Connectionistic Problem Solving: Computational Aspects of Biological Learning*, Berlin: Birkhäuser.

Hancock, P.J.B., R.J. Baddeley, and L.S. Smith, 1992. "The principal components of natural images," *Network*.

vol. 3, pp. 61–70.

Hanson, L.K., and P. Solamon, 1990. "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-12, pp. 993–1002.

Härdle, W., 1990. *Applied Nonparametric Regression*, Cambridge: Cambridge University Press.

Hardy, R.L., 1971. "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysics Research*, vol. 76, pp. 1905–1915.

Harel, D., 1987. "*Algorithmics: The Spirit of Computing*," Reading, MA: Addison-Wesley.

Hartline, H.K., 1940. "The receptive fields of optic nerve fibers," *American Journal of Physiology*, vol. 130, pp. 690–699.

Hartman, E., 1991. "A high storage capacity neural network content-addressable memory," *Network*, vol. 2, pp. 315–334.

Hartman, E.J., J.D. Keeler, and J.M. Kowalski, 1990. "Layered neural networks with Gaussian hidden units as universal approximators." *Neural Computation*, vol. 2, pp. 210–215.

Hashem, S., 1997. "Optimal linear combinations of neural networks." *Neural Networks*, vol. 10, pp. 599–614.

Hassibi, B., A.H. Sayed, and T. Kailath, 1998. *Indefinite Quadratic Estimation and Control: A Unified Approach to* H_2 *and* H_∞ *Theories*, SIAM.

Hassibi, B., A.H. Sayed, and T. Kailath, 1996. "The H^∞ optimality of the LMS algorithm," *IEEE Transactions on Signal Processing*, vol. 44, pp. 267–280.

Hassibi, B., A.H. Sayed, and T. Kailath, 1993. "LMS is H^∞ optimal," *Proceedings of the IEEE Conference on Decision and Control*, pp. 74–79, San Antonio, Texas.

Hassibi, B., D.G. Stork, and G.J. Wolff, 1992. "Optimal brain surgeon and general network pruning," *IEEE International Conference on Neural Networks*, vol. 1, pp. 293–299, San Francisco.

Hassibi, B., and T. Kailath, 1995. "H^∞ optimal training algorithms and their relation to back propagation," *Advances in Neural Information Proccessing Systems*, vol. 7, pp. 191–198.

Hastie, T., and W. Stuetzle, 1989. "Principal curves," *Journal of the American Statistical Association*, vol. 84, pp. 502–516.

Hastings, W.K., 1970. "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 87, pp. 97–109.

Haussler, D., 1988. "Quantifying inductive bias: AI learning algorithms and Valiant's learning framework," *Artificial Intelligence*, vol. 36, pp. 177–221.

Hawkins, R.D., and G.H. Bower, eds., 1989. *Computational Models of Learning in Simple Neural Systems*, San Diego, CA: Academic Press.

Haykin, S., 1996a. *Adaptive Filter Theory*, 3rd edition, Englewood Cliffs, NJ: Prentice-Hall.

Haykin, S., 1996b. "Neural networks expand SP's horizons," *IEEE Signal Processing Magazine*, vol. 13, no. 2, pp. 24–29.

Haykin, S., ed. 1994a. *Blind Deconvolution*, Englewood Cliffs, NJ: Prentice-Hall.

Haykin, S., 1994b. *Communication Systems*, 3rd edition, New York: John Wiley.

Haykin, S., 1992. "Blind equalization formulated as a self-organized learning process," *Proceedings of the Twenty-Sixth Asilomar Conference on Signals, Systems, and Computers*, pp. 346–350, Pacific Grove, CA.

Haykin, S., and C. Deng, 1991. "Classification of radar clutter using neural networks," *IEEE Transactions on Neural Networks*, vol. 2, pp. 589–600.

Haykin, S., and B. Kosko, eds. 1998. Special Issue of *Proceedings of the IEEE on Intelligent Signal Processing*, vol. 88, to appear.

Haykin, S., and J. Principe, 1998. "Making sense of a complex world: Using neural networks to dynamically model chaotic events such as sea clutter," *IEEE Signal Processing Magazine*, vol. 15, to appear.

Haykin, S., W. Stehwien, P. Weber, C. Deng, and R. Mann, 1991. "Classification of radar clutter in air traffic control environment," *Proceedings of the IEEE*, vol. 79, pp. 741–772.

Haykin, S., P. Yee, and E. Derbez, 1997. "Optimum nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2774–2786.

Haykin, S., and B. Van Veen, 1998. *Signals and Systems*, New York: Wiley.

Hebb, D.O., 1949. *The Organization of Behavior: A Neuropsychological Theory*, New York: Wiley.

Hecht-Nielsen, R., 1995. "Replicator neural networks for universal optimal source coding," *Science*, vol. 269, pp. 1860–1863.

Hecht-Nielson, R., 1990. *Neurocomputing*, Reading, MA: Addison-Wesley.

Hecht-Nielson, R., 1987. "Kolmogorov's mapping neural network existence theorem," *First IEEE International Conference on Neural Networks*, vol. III, pp. 11–14, San Diego, CA.

Helstrom, C.W., 1968. *Statistical Theory of Signal Detection*, 2nd edition, Pergamon Press.

Herault, J., and C. Jutten, 1994. *Reseaux Neuronaux et Traitement du Signal*, Paris: Hermes Publishers.

Herault, J., and C. Jutten, 1986. "Space or time adaptive signal processing by neural network models," in J.S. Denker, ed., *Neural Networks for Computing*. Proceedings of the AIP Conference, American Institute of

Physics, New York, pp. 206–211.

Herault, J., C. Jutten, and B. Ans, 1985. "Detection de grandeurs primitives dans un message composite par une architecture de calcul neuromimetique un apprentissage non supervise." *Procedures of GRETSI*, Nice, France.

Hertz, J., A. Krogh, and R.G. Palmer, 1991. *Introduction to the Theory of Neural Computation*, Reading, MA: Addison-Wesley.

Hestenes, M.R., and E. Stiefel, 1952. "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards*, vol. 49, pp. 409–436.

Hetherington, P.A., and M.L. Shapiro, 1993. "Simulating Hebb cell assemblies: The necessity for partitioned dendritic trees and a post-not-pre LTD rule," *Network*, vol. 4, pp. 135–153.

Hiller, F.S., and G.J. Lieberman, 1995. *Introduction to Operations Research*, 6th edition, New York: McGraw-Hill.

Hinton, G.E., 1989. "Connectionist learning procedures," *Artificial Intelligence*, vol. 40, pp. 185–234.

Hinton, G.E., 1989. "Deterministic Boltzmann machine learning performs steepest descent in weight-space," *Neural Computation*, vol. 1, pp. 143–150.

Hinton, G.E., 1981. "Shape representation in parallel systems," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, British Columbia.

Hinton, G.E., P. Dayan, B.J. Frey, and R.M. Neal, 1995. "The 'wake-sleep' algorithm for unsupervised neural networks," *Science*, vol. 268, pp. 1158–1161.

Hinton, G.E., and Z. Ghahramani, 1997. "Generative models for discovering sparse distributed representations," *Philosophical Transactions of the Royal Society, Series B*, vol. 352, pp. 1177–1190.

Hinton, G.E., and S.J. Nowlan, 1990. "The bootstrap Widrow-Hoff rule as a cluster-formation algorithm," *Neural Computation*, vol. 2, pp. 355–362.

Hinton, G.E., and S.J. Nowlan, 1987. "How learning can guide evolution," *Complex Systems*, vol. 1, pp. 495–502.

Hinton, G.E., and T.J. Sejnowski, 1986. "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in Microstructure of Cognition*, D.E. Rumelhart and J.L. McClelland, eds., Cambridge, MA: MIT Press.

Hinton, G.E., and T.J. Sejnowski, 1983. "Optimal perceptual inference," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 448–453, Washington, DC.

Hirsch, M.W., 1989. "Convergent activation dynamics in continuous time networks," *Neural Networks*, vol. 2, pp. 331–349.

Hirsch, M.W., 1987. "Convergence in neural nets," *First IEEE International Conference on Neural Networks*, vol. II, pp. 115–125, San Diego, CA.

Hirsch, M.W., and S. Smale, 1974. *Differential Equations, Dynamical Systems, and Linear Algebra*, New York: Academic Press.

Hochreiter, S., 1991. *Untersuchungen zu dynamischen neuronalen Netzen*, Diploma Thesis, Technische Universität München, Germany.

Hochreiter, S., and J. Schmidhuber, 1997. "LSTM can solve hard long time lag problems," *Advances in Neural Information Processing Systems*, vol. 9, pp. 473–479, Cambridge, MA: MIT Press.

Hodgkin, A.L., and A.F. Huxley, 1952. "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Journal of Physiology*, vol. 117, pp. 500–544.

Holden, S.B., and M. Niranjan, 1995. "On the practical applicability of Vapnik-Chervonenkis dimension bounds," *Neural Computation*, vol. 7, pp. 1265–1288.

Holland, J.H., 1992. *Adaptation in Natural and Artificial Systems*, Cambridge, MA: MIT Press.

Hopcroft, J., and U. Ullman, 1979. *Introduction to Automata Theory, Languages and Computation*, Reading, MA: Addison-Wesley.

Hopfield, J.J., 1995. "Pattern recognition computation using action potential timing for stimulus representation," *Nature*, vol. 376, pp. 33–36.

Hopfield, J.J., 1994. "Neurons, dynamics and computation," *Physics Today*, vol. 47, pp. 40–46, February.

Hopfield, J.J., 1987a. "Networks, Computations, Logic, and Noise," *IEEE International Conference on Neural Networks*, vol. I, pp. 107–141, San Diego, CA.

Hopfield, J.J., 1987b. "Learning algorithms and probability distributions in feed-forward and feed-back networks," *Proceedings of the National Academy of Sciences, USA*, vol. 84, pp. 8429–8433.

Hopfield, J.J., 1984. "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences, USA*, vol. 81, pp. 3088–3092.

Hopfield, J.J., 1982. "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences, USA*, vol. 79, pp. 2554–2558.

Hopfield, J.J., and D.W. Tank, 1986. "Computing with neural circuits: A model," *Science*, vol. 233, pp. 625–633.

Hopfield, J.J., and T.W. Tank, 1985. "'Neural' computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–152.

Hopfield, J.J., D.I. Feinstein, and R.G. Palmer, 1983. "'Unlearning' has a stabilizing effect in collective memo-

ries," *Nature*, vol. 304, pp. 158–159.

Horn, B.K.P., 1977. "Understanding image intensities," *Artificial Intelligence*, vol. 8, pp. 201–237.

Hornik, K., M. Stinchcombe, and H. White, 1990. "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Networks*, vol. 3, pp. 551–560.

Hornik, K., M. Stinchcombe, and H. White, 1989. "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366.

Hotteling, H., 1933. "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 417–441, 498–520.

Hubel, D.H., 1988. *Eye, Brain, and Vision*, New York: Scientific American Library.

Hubel, D.H., and T.N. Wiesel, 1977. "Functional architecture of macaque visual cortex," *Proceedings of the Royal Society*, B, vol. 198, pp. 1–59, London.

Hubel, D.H., and T.N. Wiesel, 1962. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *Journal of Physiology*, vol. 160, pp. 106–154, London.

Huber, P.J. 1985. "Projection pursuit," *Annals of Statistics*, vol. 13, pp. 435–475.

Huber, P.J., 1981. *Robust Statistics*, New York: Wiley.

Huber, P.J., 1964. "Robust estimation of a location parameter," *Annals of Mathematical Statistics*, vol. 35, pp. 73–101.

Hush, D.R, 1997. "Learning from examples: From theory to practice," Tutorial #4, *1997 International Conference on Neural Networks*, Houston, June.

Hush, D.R., and B.G. Horne, 1993. "Progress in supervised neural networks: What's new since Lippmann?" *IEEE Signal Processing Magazine*, vol. 10, pp. 8–39.

Hush, D.R., and J.M. Salas, 1988. "Improving the learning rate of back-propagation with the gradient reuse algorithm," *IEEE International Conference on Neural Networks*, vol. I, pp. 441–447, San Diego, CA.

Illingsworth, V., E.L. Glaser, and I.C. Pyle, 1989. *Dictionary of Computing*, New York: Oxford University Press.

Intrator, N., 1992. "Feature extraction using an unsupervised neural network," *Neural Computation*, vol. 4, pp. 98–107.

Jaakkola, T., and M.I. Jordan, 1996. "Computing upper and lower bounds on likelihoods in intractable networks," in E. Horwitz, ed., *Workshop on Uncertainty in Artificial Intelligence*, Portland, Or.

Jackson, E.A., 1989. *Perspectives of Nonlinear Dynamics*, vol. 1, Cambridge: Cambridge University Press.

Jackson, E.A., 1990. *Perspectives of Nonlinear Dynamics*, vol. 2, Cambridge: Cambridge University Press.

Jackson, I.R.H., 1989, "An order of convergence for some radial basis functions," *IMA Journal of Numerical Analysis*, vol. 9, pp. 567–587.

Jackson, J.D., 1975. *Classical Electrodynamics*, 2nd edition, New York: Wiley.

Jacobs, R.A., 1990. "Task Decomposition Through Computation in a Modular Connectionist Architecture," Ph.D. Thesis, University of Massachusetts.

Jacobs, R.A., 1988. "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–307.

Jacobs, R.A., and M.I. Jordan, 1993. "Learning piecewise control strategies in a modular neural network architecture," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, pp. 337–345.

Jacobs, R.A., and M.I. Jordan, 1991. "A competitive modular connectionist architecture," *Advances in Neural Information Processing Systems*, vol. 3, pp. 767–773, San Mateo, CA: Morgan Kaufmann.

Jacobs, R.A., M.I. Jordan, S.J. Nowlan, and G.E. Hinton, 1991a. "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, pp. 79–87.

Jacobs, R.A., M.I. Jordan, and A.G. Barto, 1991b. "Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks," *Cognitive Science*, vol. 15, pp. 219–250.

Jayant, N.S., and P. Noll, 1984. *Digital Coding of Waveforms*, Englewood Cliffs, NJ: Prentice-Hall.

Jaynes, E.T., 1982. "On the rationale of maximum-entropy methods," *Proceedings of the IEEE*, vol. 70, pp. 939–952.

Jaynes, E.T., 1957. "Information theory and statistical mechanics," *Physical Review*, vol. 106, pp. 620–630; "Information theory and statistical mechanic II," *Physical Review*, vol. 108, pp. 171–190.

Jazwinski, A.H., 1970. *Stochastic Processes and Filtering Theory*, New York: Academic Press.

Jelinek, F., 1997. *Statistical Methods for Speech Recognition*, Cambridge, MA: MIT Press.

Johansson, E.M., F.U. Dowla, and D.M. Goodman, 1990. "Back-propagation learning for multi-layer feed-forward neural networks using the conjugate gradient method," Report UCRL-JC-104850, Lawrence Livermore National Laboratory, CA.

Johnson, D.S., C.R. Aragon, L.A. McGeoch, and C. Schevon, 1989. "Optimization by simulated annealing: An experimental evaluation," *Operations Research*, vol. 37, pp. 865–892.

Jolliffe, I.T., 1986. *Principal Component Analysis*, New York: Springer-Verlag.

Jones, J.P., and L.A. Palmer, 1987a. "The two-dimensional spatial structure of simple receptive fields in cat

striate cortex," *Journal of Neurophysiology,* vol. 58, pp. 1187–1211.

Jones, J.P., and L.A. Palmer, 1987b. "An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex," *Journal of Neurophysiology,* vol. 58, pp. 1233–1258.

Jones, J.P., A. Steponski, and L.A. Palmer. 1987. "The two-dimensional spectral structure of simple receptive fields in cat striate cortex," *Journal of Neurophysiology,* vol. 58, pp. 1212–1232.

Jordan, M.I., 1994. "A statistical approach to decision tree modeling," *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory,* New York: ACM Press.

Jordan, M.I., 1986. "Attractor dynamics and parallelism in a connectionist sequential machine," *The Eighth Annual Conference of the Cognitive Science Society,* pp. 531–546, Amherst, MA.

Jordan, M.I., ed., 1998. *Learning in Graphical Models,* Boston: Kluwer.

Jordan, M.I., Z. Ghahramani, T.S. Jakkolla, and L.K. Saul, 1998. "An introduction to variational methods for graphical models," In M.I. Jordan, ed., *Learning in Graphical Models,* Boston: Kluwer.

Jordan, M I., and R.A. Jacobs, 1995. "Modular and Hierarchical Learning Systems," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* pp. 579–583, Cambridge, MA: MIT Press.

Jordan, M.I., and R.A. Jacobs, 1994. "Hierarchical mixtures of experts and the EM algorithm," *Neural Computation,* vol. 6, pp. 181–214.

Jordan, M.I., and R.A. Jacobs, 1992. "Hierarchies of adaptive experts," *Advances in Neural Information Processing Systems,* vol. 4, pp. 985–992, San Mateo, CA: Morgan Kaufmann.

Joseph, R.D., 1960. "The number of orthants in n-space intersected by an s-dimensional subspace," Technical Memo 8, Project PARA, Cornell Aeronautical Lab., Buffalo, N.Y.

Jutten, C., and J. Herault, 1991. "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing,* vol. 24, pp. 1–10.

Kaas, J.H., M.M. Merzenich, and H.P. Killackey, 1983. "The reorganization of somatosensory cortex following peripheral nerve damage in adult and developing mammals," *Annual Review of Neurosciences,* vol. 6, pp. 325–356.

Kailath, T., 1980. *Linear Systems,* Englewood Cliffs, NJ: Prentice-Hall.

Kailath, T., 1974. "A view of three decades of linear filtering theory," *IEEE Transactions of Information Theory,* vol. IT-20, pp. 146–181.

Kailath, T., 1971. "RKHS approach to detection and estimation problems—Part I: Deterministic signals in Gaussian noise," *IEEE Transactions of Information Theory,* vol. IT-17, pp. 530–549.

Kailath, T., 1968. "An innovations approach to least-squares estimation: Part 1. Linear filtering in additive white noise," *IEEE Transactions of Automatic Control,* vol. AC-13, pp. 646–655.

Kalman, R.E., 1960. "A new approach to linear filtering and prediction problems," *Transactions of the ASME, Journal of Basic Engineering,* vol. 82, pp. 35–45.

Kandel, E.R., and J.H. Schwartz, 1991. *Principles of Neural Science,* 3rd ed., New York: Elsevier.

Kangas, J., T. Kohonen, and J. Laaksonen, 1990. "Variants of self-organizing maps," *IEEE Transactions on Neural Networks* 1, 93–99.

Kanter, I., and H. Sompolinsky, 1987. "Associative recall of memory without errors," *Physical Review A,* vol. 35, pp. 380–392.

Kaplen, J.L., and J.A. Yorke, 1979. "Chaotic behavior of multidimensional difference equations," in H.-O Peitgen and H.-O Walker, eds., *Functional Differential Equations and Approximations of Fixed Points,* pp. 204–227, Berlin: Springer.

Kappen, H.J., and F.B. Rodriguez, 1998. "Efficient learning in Boltzmann machines using linear response theory," *Neural Computation,* vol. 10, to appear.

Karhunen, K., 1947. "Uber lineare methoden in der Wahrscheinlichkeitsrechnung," *Annales Academiae Scientiarum Fennicae, Series AI: Mathematica-Physica,* vol. 37, pp. 3–79, (Transl.: RAND Corp., Santa Monica, CA, Rep. T-131, Aug. 1960).

Karhunen, J., and J. Joutsensalo, 1995. "Generalizations of principal component analysis, optimization problems, and neural networks," *Neural Networks,* vol. 8, pp. 549–562.

Karpinski, M., and A. Macintyre, 1997. "Polynomial bounds for VC dimension of sigmoidal and general Pfaffian neuronal networks," *Journal of Computer and System Sciences,* vol. 54, pp. 169–176.

Katagiri, S., and E. McDermott, 1996. "Discriminative training—Recent progress in speech recognition," in C.H. Chen, L.F. Pau, and P.S.P. Wang, eds., *Handbook of Pattern Recognition and Computer Vision,* 2nd edition, Singapore: World Scientific Publishing.

Katz, B., 1966. *Nerve, Muscle and Synapse,* New York: McGraw-Hill.

Kawamoto, A.H., and J.A. Anderson, 1985. "A neural network model of multistable perception," *Acta Psychologica,* vol. 59, pp. 35–65.

Kawato, M., H. Hayakama, and T. Inui, 1993. "A forward-inverse optics model of reciprocal connections between visual cortical areas," *Network,* vol. 4, pp. 415–422.

Kay, J., 1992. "Feature discovery under contextual supervision using mutual information," *International Joint*

Conference on Neural Networks, vol. IV, pp. 79–84, Baltimore.

Kearns, M., 1996. "A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split," *Advances in Neural Information Processing Systems,* vol. 8, pp. 183–189, Cambridge, MA: MIT Press.

Kearns, M., and L.G. Valiant, 1989. "Cryptographic limitations on learning Boolean formulae and finite automata," *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing,* pp. 433–444, New York.

Kearns, M.J., and U.V. Vazirani, 1994. *An Introduction to Computational Learning Theory,* Cambridge, MA: MIT Press.

Kechriotis, G., E. Zervas, and E.S. Manolakos, 1994. "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Transactions on Neural Networks,* vol. 5, pp. 267–278.

Keeler, J.D., 1986. "Basins of attraction of neural network models," in *Neural Networks for Computing,* J.S. Denker, ed., pp. 259–264. New York: American Institute of Physics.

Keller, J.B., 1976. "Inverse problems," *American Mathematical Monthly,* vol. 83, pp. 107–118.

Kelso, S.R., A.H. Ganong, and T.H. Brown, 1986. "Hebbian synapses in hippocampus," *Proceedings of the National Academy of Sciences, USA,* vol. 83, pp. 5326–5330.

Kennel, M.B., R. Brown, and H.D.I. Abarbanel, 1992. "Determining minimum embedding dimension using a geometrical construction," *Physical Review A,* vol. 45, pp. 3403–3411.

Kerlirzin, P., and F. Vallet, 1993. "Robustness in multilayer perceptrons," *Neural Computation,* vol. 5, pp. 473–482.

Kirkpatrick, S., 1984. "Optimization by simulated annealing: Quantitative Studies," *Journal of Statistical Physics,* vol. 34, pp. 975–986.

Kirkpatrick, S., and D. Sherrington, 1978. "Infinite-ranged models of spin-glasses," *Physical Review,* Series B, vol. 17, pp. 4384–4403.

Kirkpatrick, S., C.D. Gelatt, Jr., and M.P. Vecchi, 1983. "Optimization by simulated annealing," *Science,* vol. 220, pp. 671–680.

Kirsch, A., 1996. *An Introduction to the Mathematical Theory of Inverse Problems,* New York: Springer-Verlag.

Kleene, S.C., 1956. "Representation of events in nerve nets and finite automata," in C.E. Shannon and J. McCarthy, eds., *Automata Studies,* Princeton, NJ: Princeton University Press.

Kmenta, J., 1971. *Elements of Econometrics,* New York: Macmillan.

Knudsen, E.I., S. duLac, and S.D. Esterly, 1987. "Computational maps in the brain," *Annual Review of Neuroscience,* vol. 10, pp. 41–65.

Koch, C., and I. Segev, eds., 1989. *Methods in Neuronal Modeling: From Synapses to Networks,* Cambridge, MA: MIT Press.

Koch, C., T. Poggio, and V. Torre, 1983. "Nonlinear interactions in a dendritic tree: Localization, timing, and role in information processing," *Proceedings of the National Academy of Sciences, USA,* vol. 80, pp. 2799–2802.

Koch, C., and B. Mathur, 1996. "Neuromorphic vision chips," *IEEE Spectrum,* vol. 33, no. 5, pp. 38–46.

Kohonen, T., 1997a. "Exploration of very large databases by self-organizing maps," *1997 International Conference on Neural Networks,* vol. I, pp. PL1–PL6, Houston.

Kohonen, T., 1997b. *Self-Organizing Maps,* 2nd edition, Berlin: Springer-Verlag.

Kohonen, T., 1996. "Emergence of invariant-feature detectors in the adaptive-subspace self-organizing maps," *Biological Cybernetics,* vol. 75, pp. 281–291.

Kohonen, T., 1993. "Physiological interpretation of the self-organizing map algorithm." *Neural Networks,* vol. 6, pp. 895–905.

Kohonen, T., 1993. "Things you haven't heard about the self-organizing map," *Proceedings of the IEEE International Conference on neural networks,* pp. 1147–1156, San Francisco.

Kohonen, T., 1990a. "The self-organizing map," *Proceedings of the Institute of Electrical and Electronics Engineers,* vol. 78, pp. 1464–1480.

Kohonen, T., 1990b. "Improved versions of learning vector quantization," *IEEE International Joint Conference on Neural Networks,* vol. I, pp. 545–550, San Diego, CA.

Kohonen, T., 1988a. "An introduction to neural computing," *Neural Networks,* vol. 1, pp. 3–16.

Kohonen, T., 1988b. *Self-Organization and Associative Memory,* 3rd edition, New York: Springer-Verlag.

Kohonen, T., 1988c. "The 'neural' phonetic typewriter." *Computer,* vol. 21, pp. 11–22.

Kohonen, T., 1986. "Learning vector quantization for pattern recognition," *Technical Report TKK-F-A601,* Helsinki University of Technology, Finland.

Kohonen, T., 1982. "Self-organized formation of topologically correct feature maps," *Biological Cybernetics,* vol. 43, pp. 59–69.

Kohonen, T., 1972. "Correlation matrix memories," *IEEE Transactions on Computers,* vol. C-21, pp. 353–359.

Kohonen, T., and E. Oja, 1976. "Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks for neuron-like elements," *Biological Cybernetics,* vol. 21, pp. 85–95.

Kohonen, T., E. Oja, O. Simula, A. Visa, and J. Kangas, 1996. "Engineering applications of the self-organizing

map," *Proceedings of the IEEE*, vol. 84, pp. 1358-1384.

Kohonen, T., E. Reuhkala, K. Mäkisara, and L. Vainio, 1976. "Associative recall of images." *Biological Cybernetics*, vol. 22, pp. 159–168.

Kohonen, T., G. Barna, and R. Chrisley, 1988. "Statistical pattern recognition with neural networks: Benchmarking studies," *IEEE International Conference on Neural Networks*, vol. I, pp. 61–68, San Diego, CA.

Kohonen, T., J. Kangas, J. Laaksonen, and K. Torkkola, 1992. "LVQ-PAK: The learning vector quantization Program Package," Helsinki University of Technology, Finland.

Koiran, P., and E.D. Sontag, 1996. "Neural networks with quadratic VC dimension," *Advances in Neural Information Processing Systems*, vol. 8, pp. 197–203, Cambridge, MA: MIT Press.

Kolen, J.F., and J.B. Pollack, 1990. "Backpropagation is sensitive to initial conditions," *Complex Systems*, vol. 4, pp. 269–280.

Kollias, S., and D. Anastassiou, 1989. "An adaptive least squares algorithm for the efficient training of artificial neural networks," *IEEE Transactions on Circuits and Systems*, vol. 36, pp. 1092–1101.

Kollias, S., and D. Anastassiou, 1988. "Adaptive training of multilayer neural networks using a least squares estimation technique," *IEEE International Conference on Neural Networks*, vol. 1, pp. 383–390, San Diego.

Kolmogorov, A.N., 1942. "Interpolation and extrapolation of stationary random sequences," translated by the Rand Corporation, Santa Monica, CA., April 1962.

Kosko, B., 1997. *Fuzzy Engineering*, Upper Saddle River, NJ: Prentice-Hall.

Kosko, B., 1992. *Neural Networks and Fuzzy Systems*, Englewood Cliffs, NJ: Prentice-Hall.

Kosko, B., 1988. "Bidirectional associative memories," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, pp. 49–60.

Kotilainen, P., 1993. "Simulations and implementations of neural networks for principal component analysis," *Electronics Lab Report 1–93*, Tampre University of Technology, Finland.

Kraaijveld, M.A., and R.P.W. Duin, 1991. "Generalization capabilities of minimal kernel-based networks," *International Joint Conference on Neural Networks*, vol. I, pp. 843–848, Seattle.

Kramer, A.H., and A. Sangiovanni-Vincentelli, 1989. "Efficient parallel learning algorithms for neural networks," *Advances in neural Information Processing Systems*, vol. 1, pp. 40–48, San Mateo, CA: Morgan Kaufmann.

Kremer, S.C., 1996. "Comments on constructive learning of recurrent neural networks: Limitations of recurrent cascade correlation and a simple solution," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1047–1049.

Kremer, S.C., 1995. "On the computational power of Elman-style recurrent networks," *IEEE Transactions on Neural Networks*, vol. 6, pp. 1000–1004.

Kreyszig, E., 1988. *Advanced Engineering Mathematics*, 6th ed., New York: Wiley.

Krishnamurthy, A.K., S.C. Ahalt, D.E. Melton, and P. Chen, 1990. "Neural networks for vector quantization of speech and images," *IEEE Journal of Selected Areas in Communications*, vol. 8, pp. 1449–1457.

Krzyżak, A., T. Linder, and G. Lugosi, 1996. "Nonparametric estimation and classification using radial basis functions," *IEEE Transactions on Neural Networks*, vol. 7, pp. 475–487.

Kuan, C.-M., and K. Hornik, 1991. "Convergence of learning algorithms with constant learning rates," *IEEE Transactions on Neural Networks*, vol. 2, pp. 484–489.

Kuan, C.-M., K. Hornik, and H. White, 1994. "A convergence result for learning in recurrent neural networks," *Neural Computation*, vol. 6, pp. 420–440.

Kuffler, S.W., J.G. Nicholls, and A.R. Martin, 1984. *From Neuron to Brain: A Cellular Approach to the Function of the Nervous System*, 2nd edition, Sunderland, MA: Sinauer Associates.

Kullback, S., 1968. *Information Theory and Statistics*, Gloucester, MA: Peter Smith.

Kung, S.Y., and K.I. Diamantaras, 1990. "A neural network learning algorithm for adaptive principal component extraction (APEX)." *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 861–864, Albuquerque.

Kushner, H.J., and D.S. Clark, 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, New York: Springer-Verlag.

Lacoume, J.L., P.O. Amblard, and P. Comon, 1997. *Statistiques d'ordre Superieur pour le Traitement du Signal*, Masson Publishers.

Lancoz, C., 1964. *Linear Differential Operators*, London: Van Nostrand.

Landau, Y.D., 1979. *Adaptive Control: The Model Reference Approach*, New York: Marcel Dekker.

Landau, L.D., and E.M. Lifshitz, 1980. *Statistical Physics: Part 1*, 3rd edition, London: Pergamon Press.

Lanford, O.E., 1981. "Strange attractors and turbulence," in H.L. Swinney and J.P. Gollub, eds., *Hydrodynamic Instabilities and the Transition to Turbulence*, New York: Springer-Verlag.

Lang, K.J., and G.E. Hinton, 1988. "The development of the time-delay neural network architecture for speech recognition," Technical Report CMU-CS-88-152, Carnegie-Mellon University, Pittsburgh, PA.

Lapedes, A., and R. Farber, 1986. "Programming a massively parallel, computation universal system: Static

Behavior," In *Neural Networks for Computing*, J.S. Denker, ed., pp. 283–298, New York: American Institute of Physics.

Larson, J., and G. Lynch, 1989. "Theta pattern stimulation and the induction of LTP: The sequence in which synapses are stimulated determines the degree to which they potentiate," *Brain Research*, vol. 489, pp. 49–58.

LaSalle, J., and S. Lefschetz, 1961. *Stability by Liapunov's Direct Method with Applications*, New York: Academic Press.

LeCun, Y., 1993. *Efficient Learning and Second-order Methods, A Tutorial at NIPS 93*, Denver

LeCun, Y., 1989. "Generalization and network design strategies," Technical Report CRG-TR-89-4, Department of Computer Science, University of Toronto, Canada.

LeCun, Y., 1985. "Une procedure d'apprentissage pour reseau a seuil assymetrique," *Cognitiva*, vol. 85, pp. 599–604.

LeCun, Y., and Y. Bengio, 1995. "Convolutional networks for images, speech, and time series," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, Cambridge, MA: MIT Press.

LeCun, Y., B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel, 1990a. "Handwritten digit recognition with a back-propagation network," *Advances in Neural Information Processing*, vol. 2, pp. 396–404, San Mateo, CA: Morgan Kaufmann.

LeCun, Y., L. Bottou, and Y. Bengio, 1997. "Reading checks with multilayer graph transformeer networks," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 151–154, Munich, Germany.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner, 1998. "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, to appear.

LeCun, Y., J.S. Denker, and S.A. Solla, 1990. "Optimal brain damage," *Advances in Neural Information Processing Systems*, vol. 2, pp. 598–605, San Mateo, CA: Morgan Kaufmann.

LeCun, Y., I. Kanter, and S.A. Solla, 1991. "Second order properties of error surfaces: Learning time and generalization," *Advances in Neural Information Processing Systems*, vol. 3, pp. 918–924, Cambridge, MA: MIT Press.

Lee, D.D., and H.S. Seung, 1997. "Unsupervised learning by convex and conic coding," *Advances in Neural Information Processing Systems*, vol. 9, pp. 515–521, Cambridge, MA: MIT Press.

Lee, T., 1997. *Independent Component Analysis: Theory and Applications*, Ph.D. Thesis, Technische Universität, Berlin, Germany.

Lee, T.-C., A.M. Peterson and J.J-C. Tsai, 1990. "A multilayer feed-forward neural network with dynamically adjustable structures," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 367–369, Los Angeles.

Lee, Y., and R.P. Lippmann, 1990. "Practical characteristics of neural networks and conventional pattern classifiers on artificial and speech problems," *Advances in Neural Information Processing Systems*, vol. 2, pp. 168–177, San Mateo, CA: Morgan Kaufmann.

Lee, Y., S. Oh, and M. Kim, 1991. "The effect of initial weights on premature saturation in back-propagation learning," *International Joint Conference on Neural Networks*, vol. I, pp. 765–770, Seattle.

Lee, Y.C., G. Doolen, H.H. Chan, G.Z. Sen, T. Maxwell, H.Y Lee, and C.L. Giles, 1986. "Machine learning using a higher order correlation network," *Physica*, D22, pp. 276–289.

Lefebvre, W.C., 1991. *An Object Oriented Approach for the Analysis of Neural Networks*, Master's Thesis, University of Florida, Gainsville, Fl.

Leon-Garcia, A., 1994. *Probability and Random Processes for Electrical Engineering*, 2nd edition, Reading, MA: Addison-Wesley.

Leontaritis, I., and S. Billings, 1985. "Input–output parametric models for nonlinear systems: Part I: Deterministic nonlinear systems," *International Journal of Control*, vol. 41, pp. 303–328.

Levin, A.V., and K.S. Narendra, 1996. "Control of nonlinear dynamical systems using neural networks—Part II: Observability, identification, and control," *IEEE Transactions on Neural Networks*, vol. 7, pp. 30–42.

Levin, A.V., and K.S. Narendra, 1993. "Control of nonlinear dynamical systems using neural networks—Controllability and stabilization," *IEEE Transactions on Neural Networks*, vol. 4, pp. 192–206.

Levine, M., 1985. *Man and Machine Vision*, New York: McGraw-Hill.

Lewis, F.L., and V.L. Syrmas, 1995. *Optimal Control*, 2nd edition, New York: Wiley (Interscience).

Lewis, F.L., A. Yesildirek, and K. Liu, 1996. "Multilayer neural-net robot controller with guaranteed tracking performance," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1–12.

Lichtenberg, A.J., and M.A. Lieberman, 1992. *Regular and Chaotic Dynamics*, 2nd edition, New York: Springer-Verlag.

Light, W.A., 1992a. "Some aspects of radial basis function approximation," in *Approximation Theory, Spline Functions and Applications*, S.P. Singh, ed., NATO ASI vol. 256, pp. 163–190, Boston: Kluwer Academic Publishers.

Light, W., 1992b. "Ridge functions, sigmoidal functions and neural networks," in E.W. Cheney, C.K. Chui, and

L.L. Schumaker, eds., *Approximation Theory VII*, pp. 163–206, Boston: Academic Press.

Lin, J.K., D.G. Grier, and J.D. Cowan, 1997. "Faithful representation of separable distributions," *Neural Computation*, vol. 9, pp. 1305–1320.

Lin, S., 1965. "Computer solutions of the traveling salesman problem," *Bell System Technical Journal*, vol. 44, pp. 2245–2269.

Lin, T., B.G. Horne, P. Tino, and C.L. Giles, 1996. "Learning long-term dependencies in NARX recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1329–1338.

Linde, Y., A. Buzo, and R. M. Gray, 1980. "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. COM-28, pp. 84–95.

Linsker, R., 1993. "Deriving receptive fields using an optimal encoding criterion," *Advances in Neural Information Processing Systems*, vol. 5, pp. 953–960, San Mateo, CA: Morgan Kaufmann.

Linsker, R., 1990a. "Designing a sensory processing system: What can be learned from principal components analysis?" *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, pp. 291–297, Washington, DC.

Linsker, R., 1990b. "Self-organization in a perceptual system: How network models and information theory may shed light on neural organization," Chapter 10 in *Connectionist Modeling and Brain Function: The Developing Interface*, S.J. Hanson and C.R. Olson, eds., pp. 351–392, Cambridge, MA: MIT Press.

Linsker, R., 1990c. "Perceptual neural organization: Some approaches based on network models and information theory," *Annual Review of Neuroscience*, vol. 13, pp. 257–281.

Linsker, R., 1989a. "An application of the principle of maximum information preservation to linear systems," *Advances in Neural Information Processing Systems*, vol. 1, pp. 186–194, San Mateo, CA: Morgan Kaufmann.

Linsker, R., 1989b. "How to generate ordered maps by maximizing the mutual information between input and output signals," *Neural computation*, vol. 1, pp. 402–411.

Linsker, R., 1988a. "Self-organization in a perceptual network," *Computer*, vol. 21, pp. 105–117.

Linsker, R., 1988b. "Towards an organizing principle for a layered perceptual network," in *Neural Information Processing Systems*, D.Z. Anderson, ed., pp. 485–494, New York: American Institute of Physics.

Linsker, R., 1987. "Towards an organizing principle for perception: Hebbian synapses and the principle of optimal neural encoding," *IBM Research Report RC12820*, IBM Research, Yorktown Heights, NY.

Linsker, R., 1986. "From basic network principles to neural architecture" (series), *Proceedings of the National Academy of Sciences, USA*, vol. 83, pp. 7508–7512, 8390–8394, 8779–8783.

Lippmann, R.P., 1987. "An introduction to computing with neural nets," *IEEE ASSP Magazine*, vol. 4, pp. 4–22.

Lippmann, R.P., 1989a. "Review of neural networks for speech recognition," *Neural Computation*, vol. 1, pp. 1–38.

Lippmann, R.P., 1989b. "Pattern classification using neural networks," *IEEE Communications Magazine*, vol. 27, pp. 47–64.

Little, W.A., 1974. "The existence of persistent states in the brain," *Mathematical Biosciences*, vol. 19, pp. 101–120.

Little, W.A., and G.L. Shaw, 1978. "Analytic study of the memory storage capacity of a neural network," *Mathematical Biosciences*, vol. 39, pp. 281–290.

Little, W.A., and G.L. Shaw, 1975. "A statistical theory of short and long term-memory," *Behavioral Biology*, vol. 14, pp. 115–133.

Livesey, M., 1991. "Clamping in Boltzmann machines," *IEEE Transactions on Neural Networks*, vol. 2, pp. 143–148.

Ljung, L., 1987. *System Identification: Theory for the User*. Englewood Cliffs, NJ: Prentice-Hall.

Ljung, L., 1977. "Analysis of recursive stochastic algorithms," *IEEE Transactions on Automatic Control*, vol. AC-22, pp. 551–575.

Ljung, L., and T. Glad, 1994. *Modeling of Dynamic Systems*, Englewood Cliffs, NJ: Prentice-Hall.

Lloyd, S.P., 1957. "Least squares quantization in PCM," unpublished Bell Laboratories technical note. Published later under the same title in *IEEE Transactions on Information Theory*, vol. IT-28, pp. 127–135, 1982.

Lo, Z.-P., M. Fujita, and B. Bavarian, 1991. "Analysis of neighborhood interaction in Kohonen neural networks," *6th International Parallel Processing Symposium Proceedings*, pp. 247–249, Los Alamitos, CA.

Lo, Z.-P., Y. Yu and B. Bavarian, 1993. "Analysis of the convergence properties of topology preserving neural networks," *IEEE Transactions on Neural Networks*, vol. pp. 207–220.

Lockery, S.R., Y. Fang, and T.J. Sejnowski, 1990. "A dynamical neural network model of sensorimotor transformations in the leech," *International Joint Conference on Neural Networks*, vol. I, pp. 183–188, San Diego, CA.

Loéve, M., 1963. *Probability Theory*, 3rd edition, New York: Van Nostrand

Lorentz, G.G., 1976. "The 13th problem of Hilbert," *Proceedings of Symposia in Pure Mathematics*, vol. 28, pp. 419–430.

Lorentz, G.G., 1966. *Approximation of Functions*, Orlando, FL: Holt, Rinehart & Winston.

Lorenz, E.N., 1963. "Deterministic non-periodic flows," *Journal of Atmospheric Sciences*, vol. 20, pp. 130–141.

Lowe, D., 1989. "Adaptive radial basis function nonlinearities, and the problem of generalisation," *First IEE International Conference on Artificial Neural Networks*, pp. 171–175, London.

Lowe, D., 1991a. "What have neural networks to offer statistical pattern processing?" *Proceedings of the SPIE Conference on Adaptive Signal Processing*, pp. 460–471, San Diego, CA.

Lowe, D., 1991b. "On the iterative inversion of RBF networks: A statistical interpretation," *Second IEE International Conference on Artificial Neural Networks*, pp. 29–33, Bournemouth, England.

Lowe, D., and A.R. Webb, 1991a. "Time series prediction by adaptive networks: A dynamical systems perspective," *IEE Proceedings (London), Part F*, vol. 138, pp. 17–24.

Lowe, D., and A.R. Webb, 1991b. "Optimized feature extraction and the Bayes decision in feed-forward classifier networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13, 355–364.

Lowe, D., and A.R. Webb, 1990. "Exploiting prior knowledge in network optimization: an illustration from medical prognosis," *Network*, vol. 1, pp. 299–323.

Lowe, D., and M.E. Tipping, 1996. "Neuroscale: Novel topographic feature extraction using RBF networks," *Neural Information Processing Systems*, vol. 9, pp. 543–549, Cambridge, MA: MIT Press.

Luenberger, D.G., 1984. *Linear and Nonlinear Programming*, 2nd edition, Reading, MA: Addison-Wesley.

Lui, H.C., 1990. "Analysis of decision contour of neural network with sigmoidal nonlinearity," *International Joint Conference on Neural Networks*, vol. I, pp. 655–659, Washington, DC.

Luo, Z., 1991. "On the convergence of the LMS algorithm with adaptive learning rate for linear feedforward networks," *Neural Computation*, vol. 3, pp. 226–245.

Luo, F., and R. Unbehauen, 1997. *Applied Neural Networks for Signal Processing*, New York: Cambridge University Press.

Luttrell, S.P., 1997. "A unified theory of density models and auto-encoders," *Technical Report 97303*, Defence Research Agency, Great Malvern, UK.

Luttrell, S.P., 1994. "A Bayesian analysis of self-organizing maps," *Neural Computation*, vol. 6, pp. 767–794.

Luttrell, S.P., 1991a. "Code vector density in topographic mappings: Scalar case," *IEEE Transactions on Neural Networks*, vol. 2, pp. 427–436.

Luttrell, S.P., 1991b. "Self-supervised training of hierarchical vector quantizers," *2nd International Conference on Artificial Neural Networks*, pp. 5–9, Bournemouth, England.

Luttrell, S.P., 1989a. "Hierarchical vector quantization," *IEE Proceedings (London)*, vol. 136 (Part I), pp. 405–413.

Luttrell, S.P., 1989b. "Self-organization: A derivation from first principle of a class of learning algorithms," *IEEE Conference on Neural Networks*, pp. 495–498, Washington, DC.

Maass, W., 1993. "Bounds for the computational power and learning complexity of analog neural nets," *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pp. 335–344, New York: ACM Press.

Maass. W., 1993. "Vapnik–Chervonenkis dimension of neural networks," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, Cambridge, MA: MIT Press.

Mach, E., 1865. "Über die Wirkung der räumlichen Vertheilung des Lichtreizes auf die Netzhaut, I. Sitzungsberichte der Mathematisch-Naturwissenschaftslichen Klasse der Kaiserlichen Akademie der Wissenschaften," vol. 52, pp. 303–322.

MacKay, D., 1992a. "Bayesian interpolation," *Neural Computation*, vol. 4, pp. 415–447.

MacKay, D., 1992b. "A practical Bayesian framework for back-propagation networks," *Neural Computation*, vol. 4, pp. 448–472.

MacKay, D.J.C., and K.D. Miller, 1990. "Analysis of Linsker's simulations of Hebbian rules," *Neural Computation*, vol. 2, pp. 173–187.

Macintyre, A.J., and E.D. Sontag, 1993. "Fitness results for sigmoidal 'neuronal' networks," *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pp. 325–334, New York: ACM Press.

MacQueen, J., 1967. "Some methods for classification and analysis of multivariate observation," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, L.M. LeCun and J. Neyman, eds., vol. 1, pp. 281–297, Berkeley: University of California Press.

Madhuranath, H., and S. Haykin, 1998. "Improved Activation Functions for Blind Separation: Details of Algebraic Derivations," *CRL Internal Report No. 358*, Communications Research Laboratory, McMaster University, Hamilton, Ontario.

Mahowald, M.A., and C. Mead, 1989. "Silicon retina," in *Analog VLSI and Neural Systems* (C. Mead), Chapter 15. Reading, MA: Addison-Wesley.

Mandelbrot, B.B., 1982. *The Fractal Geometry of Nature*, San Francisco: Freeman.

Mañé, R., 1981. "On the dimension of the compact invariant sets of certain non-linear maps," in D. Rand and L.S. Young, eds., *Dynamical Systems and Turbulence*, Lecture Notes in Mathematics, vol. 898, pp. 230–242, Berlin: Springer-Verlag.

Marr, D., 1982. *Vision*, New York: W.H. Freeman and Company.

Martinetz, T.M., H.J. Ritter, and K.J. Schulten, 1990. "Three-dimensional neural net for learning visuomotor coordination of a robot arm," *IEEE Transactions on Neural Networks*, vol. 1, pp. 131–136.

Mason, S.J., 1953. "Feedback theory—Some properties of signal-flow graphs," *Proceedings of the Institute of Radio Engineers*, vol. 41, pp. 1144–1156.

Mason, S.J., 1956. "Feedback theory—Further properties of signal-flow graphs," *Proceedings of the Institute of Radio Engineers*, vol. 44, pp. 920–926.

Maybeck, P.S., 1982. *Stochastic Models, Estimation, and Control*, vol. 2, New York: Academic Press.

Maybeck, P.S., 1979. *Stochastic Models, Estimation, and Control*, vol. 1, New York: Academic Press.

Mazaika, P.K., 1987. "A mathematical model of the Boltzmann machine," *IEEE First International Conference on Neural Networks*, vol. III, pp. 157–163, San Diego, CA.

McBride, L.E., Jr., and K.S. Narendra, 1965. "Optimization of time-varying systems," *IEEE Transactions on Automatic Control*, vol. AC-10, pp. 289–294.

McCullagh, P., and J.A. Nelder, 1989. *Generalized Linear Models*, 2nd edition, London: Chapman and Hall.

McCulloch, W.S., 1988. *Embodiments of Mind*, Cambridge, MA: MIT Press.

McCulloch, W.S., and W. Pitts, 1943. "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133.

McEliece, R.J., E.C. Posner, E.R. Rodemich, and S.S. Venkatesh, 1987. "The capacity of the Hopfield associative memory," *IEEE Transactions on Information Theory*, vol. IT-33, pp. 461–482.

McLachlan, G.J., and K.E. Basford, 1988. *Mixture Models: Inference and Applications to Clustering*, New York: Marcel Dekker.

McLachlan, G.J., and T. Krishnan, 1997. *The EM Algorithm and Extensions*, New York: Wiley (Interscience).

McQueen, J., 1967. "Some methods for classification and analysis of multivariate observations," *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, Berkeley, CA: University of California Press.

Mead, C.A., 1990. "Neuromorphic electronic systems," *Proceedings of the Institute of Electrical and Electronics Engineers*, vol. 78, pp. 1629–1636.

Mead, C.A., 1989. *Analog VLSI and Neural Systems*, Reading, MA: Addison-Wesley.

Mead, C.A., and M.A. Mahowald, 1988. "A silicon model of early visual processing," *Neural Networks*, vol. 1, pp. 91–97.

Mead, C.A., X. Arreguit, and J. Lazzaro, 1991. "Analog VLSI model of binaural hearing," *IEEE Transactions on Neural Networks*, vol. 2, pp. 232–236.

Mecklenbräuker, W., and F. Hlawatsch, eds., 1997. *The Wigner Distribution*, New York: Elsevier.

Memmi, D., 1989. "Connectionism and artificial intelligence," *Neuro-Nimes'89 International Workshop on Neural Networks and their Applications*, pp. 17–34, Nimes, France.

Mendel, J.M., 1995. *Lessons in Estimation Theory for Signal Processing, Communications, and Control*. Englewood Cliffs, NJ: Prentice-Hall.

Mendel, J.M., and R.W. McLaren, 1970. "Reinforcement-learning control and pattern recognition systems," in *Adaptive, Learning, and Pattern Recognition Systems: Theory and Applications*, vol. 66, J.M. Mendel and K.S. Fu, eds., pp. 287–318, New York: Academic Press.

Mennon, A., K. Mehrotra, C.K. Mohan, and S. Ranka, 1996. "Characterization of a class of sigmoid functions with applications to neural networks," *Neural Networks*, vol. 9, pp. 819–835.

Mercer, J., 1909. "Functions of positive and negative type, and their connection with the theory of integral equations," *Transactions of the London Philosophical Society (A)*, vol. 209, pp. 415–446.

Mesulam, M.M., 1985. "Attention, confusional states, and neglect," in *Principles of Behavioral Neurology*, M.M. Mesulam, ed., Philadelphia: F.A. Davis.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, 1953. Equations of state calculations by fast computing machines, *Journal of Chemical Physics*, vol. 21, pp. 1087–1092.

Mhaskar, H.N., 1996. "Neural networks for optimal approximation of smooth and analytic functions," *Neural Computation*, vol. 8, pp. 1731–1742.

Mhaskar, H.N., and C.A. Micchelli, 1992. "Approximation by superposition of sigmoidal and radial basis functions," *Advances in Applied Mathematics*, vol. 13, pp. 350–373.

Micchelli, C.A., 1986. "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constructive Approximation*, vol. 2, pp. 11–22.

Miller, D., A.V. Rao, K. Rose, and A. Gersho, 1996. "A global optimization technique for statistical classifier design," *IEEE Transactions on Signal Processing*, vol. 44, pp. 3108–3122.

Miller, K.D., J.B. Keller, and M.P. Stryker, 1989. "Ocular dominance column development: Analysis and simulation," *Science*, vol. 245, pp. 605–615.

Miller, D., and K. Rose, 1996. "Hierarchical, unsupervised learning with growing via phase transitions," *Neural Computation*, vol. 8, pp. 425–450.

Miller, D., and K. Rose, 1994. "Combined source-channel vector quantization using deterministic annealing,"

IEEE Transactions on Communications, vol. 42, pp. 347–356.

Miller, R., 1987. "Representation of brief temporal patterns, Hebbian synapses, and the left-hemisphere dominance for phoneme recognition," *Psychobiology*, vol. 15, pp. 241–247.

Minai, A.A., and R.J. Williams, 1990. "Back-propagation heuristics: A study of the extended delta-bar-delta algorithm," *IEEE International Joint Conference on Neural Networks*, vol. I, pp. 595–600, San Diego, CA.

Minsky, M.L., 1986. Society of Mind, New York: Simon and Schuster.

Minsky, M.L., 1967. *Computation: Finite and Infinite Machines*. Englewood Cliffs, NJ: Prentice-Hall.

Minsky, M.L., 1961. "Steps towards artificial intelligence," *Proceedings of the Institute of Radio Engineers*, vol. 49, pp. 8–30 (Reprinted in: Feigenbaum, E.A., and J. Feldman, eds., *Computers and Thought*, pp. 406–450, New York: McGraw-Hill.)

Minsky, M.L., 1954. "Theory of neural-analog reinforcement systems and its application to the brain-model problem," Ph.D. thesis, Princeton University, Princeton, NJ.

Minsky, M.L., and S.A. Papert, 1988. *Perceptrons*, expanded edition, Cambridge, MA: MIT Press.

Minsky, M.L., and S.A. Papert, 1969. *Perceptrons*, Cambridge, MA: MIT Press.

Minsky, M.L., and O.G. Selfridge, 1961. "Learning in random nets," *Information Theory, Fourth London Symposium*, London: Butterworths.

Mitchell, T.M., 1997. *Machine Learning*. New York: McGraw-Hill.

Mitchison, G., 1989. "Learning algorithms and networks of neurons," in *The Computing Neuron* (R. Durbin, C. Miall, and G. Michison, eds), pp. 35–53, Reading, MA: Addison-Wesley.

Møller, M.F., 1993. "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, pp. 525–534.

Moody, J., and C.J. Darken, 1989. "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281–294.

Moody, J., and L. Wu, 1996. "Optimization of trading systems and portfolios," in A. Weigend, Y. Abu-Mostafa, and A.-P.N. Refenes, eds., *Decision Technologies for Financial Engineering*, pp. 23–35, Singapore: World Scientific.

Moody, J.E., and T. Rögnvaldsson, 1997. "Smoothing regularizers for projective basis function networks," *Advances in Neural Information Processing Systems*, vol. 9, pp. 585–591.

Moray, N., 1959. "Attention in dichotic listening: Affective cues and the influence of instructions," *Quarterly Journal of Experimental Psychology*, vol. 27, pp. 56–60.

Morgan, N., and H. Bourlard, 1990. "Continuous speech recognition using multilayer perceptrons with hidden Markov models," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 413–416, Albuquerque.

Morita, M., 1993. "Associative memory with nonmonotonic dynamics," *Neural Networks*, vol. 6, pp. 115–126.

Morozov, V.A., 1993. *Regularization Methods for Ill-Posed Problems*, Boca Raton, FL: CRC Press.

Morse, P.M., and H. Feshbach, 1953. *Methods of Theoretical Physics*, Part 1, New York: McGraw-Hill.

Mozer, M.C., 1994. "Neural net architectures for temporal sequence processing," in A.S. Weigend and N.A. Gershenfeld, eds., *Time Series Prediction: Forecasting the Future and Understanding the Past*, pp. 243–264, Reading, MA: Addison-Wesley.

Mpitsos, G.J., 1990. "Chaos in brain function and the problem of nonstationarity: A commentary," in *Chaos in Brain Function*, E. Basar, ed., pp. 162–176. New York: Springer-Verlag.

Müller, B., and J. Reinhardt, 1990. *Neural Networks: An Introduction*, New York: Springer-Verlag.

Muller, D., and G. Lynch, 1988. "Long-term potentiation differentially affects two components of synaptic responses in hippocampus," *Proceedings of the National Academy of Sciences*, USA, vol. 85, pp. 9346–9350.

Mumford, D., 1994. "Neural architectures for pattern-theoretic problems," in C. Koch and J. Davis, eds., *Large-Scale Theories of the Cortex*, pp. 125–152, Cambridge, MA: MIT Press.

Murray, M.K., and J.W. Rice, 1993. *Differential Geometry and Statistics*, New York: Chapman and Hall.

Murtagh, B., and M. Saunders, 1978. "Large-scale linearly constrained optimization," *Mathematical Programming*, vol. 14, pp. 41–72.

Muselli, M., 1997. "On convergence properties of pocket algorithm," *IEEE Transactions on Neural Networks*, vol. 8, pp. 623–629.

Nadal, J.-P., and N. Parga, 1997. "Redundancy reduction and independent component analysis: Conditions on cumulants and adaptive approaches," *Neural Computation*, vol. 9, pp. 1421–1456.

Nadal, J.-P., and N. Parga, 1994. "Nonlinear neurons in the low-noise limit: A factorial code maximizes information transfer," *Network*, vol. 5, pp. 565–581.

Nadaraya, É.A., 1965. "On nonparametric estimation of density functions and regression curves," *Theory of Probability and its Applications*, vol. 10, pp. 186–190.

Nadaraya, É.A., 1964. "On estimating regression," *Theory of Probability and its Applications*, vol. 9, pp. 141–142.

Naftaly, U., N. Intrator, and D. Horn, 1997. "Optimal ensemble averaging of neural networks," *Network*, vol. 8, pp. 283–296.

Nakano, K., 1972. "Association—a model of associative memory," *IEEE Transactions on Systems, Man, and*

Cybernetics, vol. SMC-2, pp. 380–388.

Narendra, K.S., 1995. *Neural Networks for Identification and Control,* NIPS 95, Tutorial Program, pp. 1–46, Denver.

Narendra, K.S., and A.M. Annaswamy, 1989. *Stable Adaptive Systems,* Englewood Cliffs, NJ: Prentice-Hall.

Narendra, K.S., and K. Parthasarathy, 1990. "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks,* vol. 1, pp. 4–27.

Natarajan, B.K., 1991. *Machine Learning: A Theoretical Approach,* San Mateo, CA: Morgan Kaufmann.

Neal, R.M., 1995. *Bayesian Learning for Neural Networks,* Ph.D. Thesis, University of Toronto, Canada.

Neal, R.M., 1993. "Bayesian learning via stochastic dynamics," *Advances in Neural Information Processing Systems,* vol. 5, pp. 475–482, San Mateo, CA: Morgan Kaufmann.

Neal, R.M., 1992. "Connectionist learning of belief networks," *Artificial Intelligence,* vol. 56, pp. 71–113.

Newcomb, S., 1886. "A generalized theory of the combination of observations so as to obtain the best result," *American Journal of Mathematics,* vol. 8, pp. 343–366.

Newell, A., and H.A. Simon, 1972. *Human Problem Solving,* Englewood Cliffs, NJ: Prentice-Hall.

Ng, K., and R.P. Lippmann, 1991. "Practical characteristics of neural network and conventional pattern classifiers," *Advances in Neural Information Processing Systems,* vol. 3, pp. 970–976, San Mateo, CA: Morgan Kaufmann.

Nguyen, D., and B. Widrow, 1989. "The truck backer-upper: An example of self-learning in neural networks," *International Joint Conference on Neural Networks,* vol. II, pp. 357–363, Washington, DC.

Nie, J., and S. Haykin, 1998. "A Q-learning-based dynamic channel assignment technique for mobile communication systems," *IEEE Transactions on Vehicular Technology,* to appear.

Nie, J., and S. Haykin, 1996. "A dynamic channel assignment policy through Q-learning," *CRL Report No. 334,* Communications Research Laboratory, McMaster University, Hamilton, Ontario.

Nilsson, N.J., 1980. *Principles of Artificial Intelligence,* New York: Springer-Verlag.

Nilsson, N.J., 1965. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems,* New York: McGraw-Hill.

Niyogi, P., and F. Girosi, 1996. "On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions," *Neural Computation,* vol. 8, pp. 819–842.

Novikoff, A.B.J., 1962. "On convergence proofs for perceptrons," in *Proceedings of the Symposium on the Mathematical Theory of Automata,* pp. 615–622, Brooklyn, NY: Polytechnic Institute of Brooklyn.

Nowlan, S.J., 1990. "Maximum likelihood competitive learning," *Advances in Neural Information Processing Systems,* vol. 2, pp. 574–582, San Mateo, CA: Morgan Kaufmann.

Nowlan, S.J., and G.E. Hinton, 1992. "Adaptive soft weight tying using Gaussian mixtures," *Advances in Neural Information Processing Systems,* vol. 4, pp. 993–1000, San Mateo, CA: Morgan Kaufmann.

Nowlan, S.J., and G.E. Hinton, 1991. "Evaluation of adaptive mixtures of competing experts," *Advances in Neural Information Processing Systems,* vol. 3, pp. 774–780, San Mateo, CA: Morgan Kaufmann.

Obermayer, K., H. Ritter, and K. Schulten, 1991. "Development and spatial structure of cortical feature maps: A model study," *Advances in Neural Information Processing Systems,* vol. 3, pp. 11–17, San Mateo, CA: Morgan Kaufmann.

Oja, E., 1992a. "Principal components, minor components, and linear neural networks," *Neural Networks,* vol. 5, 927–936.

Oja, E., 1992b. "Self-organizing maps and computer vision," in *Neural Networks for Perception,* vol. 1, H. Wechsler, ed., vol. 1, pp. 368–385, San Diego, CA: Academic Press.

Oja, E., 1991. "Data compression, feature extraction, and autoassociation in feedforward neural networks," *Artificial Neural Networks,* vol. 1, pp. 737–746, Amsterdam: North-Holland.

Oja, E., 1989. "Neural networks, principal components, and subspaces," *International Journal of Neural Systems,* vol. 1, 61–68.

Oja, E., 1983. *Subspace Methods of Pattern Recognition,* Letchworth, England: Research Studies Press.

Oja, E., 1982. "A simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology,* vol. 15, pp. 267–273.

Oja, E., and J. Karhunen, 1985. "A stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *Journal of Mathematical Analysis and Applications,* vol. 106, pp. 69–84.

Oja, E., and T. Kohonen, 1988. "The subspace learning algorithm as formalism for pattern recognition and neural networks," *IEEE International Conference on Neural Networks,* vol. I, pp. 277–284, San Diego, CA.

Omlin, C.W., and C.L. Giles, 1996. "Constructing deterministic finite-state automata in recurrent neural networks," *Journal of the Association for Computing Machinery,* vol. 43, pp. 937–972.

Oppenheim, A.V., and R.W. Schafer, 1989. *Discrete-Time Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall.

Orlando, J., R. Mann, and S. Haykin, 1990. "Classification of sea-ice using a dual-polarized radar," *IEEE Journal of Oceanic Engineering,* vol. 15, pp. 228–237.

Osherson, D.N., S. Weinstein, and M. Stoli, 1990. "Modular learning," *Computational Neuroscience*, E.L. Schwartz, ed., pp. 369–377, Cambridge, MA: MIT Press.

Osuna, E., 1998. "Support Vector Machines: Training and Applications," Ph.D. Thesis, Operations Research Center, MIT.

Osuna, E., and F. Girosi, 1998. "Reducing the run-time complexity of support vector machines," ICPR 98, Brisbane, Australia.

Osuna, E., R. Freund, and F. Girosi, 1997. "An improved training algorithm for support vector machines," *Neural Networks for Signal Processing* VII, Proceedings of the 1997 IEEE Workshop, pp. 276–285, Amelia Island, FL.

Ott, E., 1993. *Chaos in Dynamical Systems*, Cambridge, MA: Cambridge University Press.

Packard, N.H., J.P. Crutchfield, J.D. Farmer, and R.S. Shaw, 1980. "Geometry from a time series," *Physical Review Letters*, vol. 45, pp. 712–716.

Palm, G., 1982. *Neural Assemblies: An Alternative Approach*, New York: Springer-Verlag.

Palmieri, F., and S.A. Shah, 1990. "Fast training of multilayer perceptrons using multi-linear parameterization," *International Joint Conference on Neural Networks*, vol. I, pp. 696–699, Washington, DC.

Palmieri, F., J. Zhu, and C. Chang, 1993. "Anti-Hebbian learning in topologically constrained linear networks: A tutorial," *IEEE Transactions on Neural Networks*, vol. 5, pp. 748–761.

Papoulis, A., 1984. *Probability, Random Variables, and Stochastic Processes*, 2nd edition, New York: McGraw-Hill.

Parisi, G., 1988. *Statistical Field Theory*, Reading, MA: Addison-Wesley.

Park, J., and I.W. Sandberg, 1991. "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, pp. 246–257.

Parker, D.B., 1987. "Optimal algorithms for adaptive networks: Second order back propagation, second order direct propagation, and second order Hebbian learning," *IEEE 1st International Conference on Neural Networks*, vol. 2, pp. 593–600, San Diego, CA.

Parker, D.B., 1985. "Learning-logic: Casting the cortex of the human brain in silicon," *Technical Report TR-47*, Center for Computational Research in Economics and Management Science, Cambridge, MA: MIT Press.

Parker, T.S., and L.O., Chua, 1989. *Practical Numerical Algorithms for Chaotic Systems*, New York: Springer.

Parzen, E., 1962. "On estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076.

Passino, K.N., 1996. "Toward bridging the perceived gap between conventional and intelligent control," in M.D. Gupta and N.K. Sinha, eds., *Intelligent Control Systems*, pp. 3–27, New York: IEEE Press.

Pavlov, I.P., 1927. *Conditional Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex*, (Translation from the Russian by G.V. Anrep), New York: Oxford University Press.

Pearl, J., 1988. *Probabilistic Reasoning in Intelligent Systems*, San Mateo, CA: Morgan Kaufmann. (Revised 2nd printing, 1991).

Pearlmutter, B.A., 1989. "Learning state-space trajectories in recurrent neural networks," *Neural Computation*, vol. 1, pp. 263–269.

Pearson, K., 1901. "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, pp. 559–572.

Peretto, P. 1984. "Collective properties of neural networks: A statistical physics approach," *Biological Cybernetics*, vol. 50, pp. 51–62.

Peretto, P., and J.-J Niez, 1986. "Stochastic dynamics of neural networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-16, pp. 73–83.

Perrin, D., 1990. "Finite automata," in J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, Chapter 1, pp. 3–57, Cambridge, MA: MIT Press.

Perrone, M.P., 1993. "Improving regression estimation: Averaging methods for variance reduction with extensions, to general convex measure optimization," Ph.D. Thesis, Brown University, Rhode Island.

Personnaz, L., I. Guyon, and G. Dreyfus, 1985. "Information storage and retrieval in spin-glass like neural networks," *Journal of Physique, Letters*, Orsay, France, vol. 46, L-359–L-365.

Peterson, C., 1991. "Mean field theory neural networks for feature recognition, content addressable memory and optimization," *Connection Science*, vol. 3, pp. 3–33.

Peterson, C., and J.R. Anderson, 1987. "A mean field theory learning algorithm for neural networks," *Complex Systems*, vol. 1, pp. 995–1019.

Peterson, C., and E. Hartman, 1989. "Explorations of the mean field theory learning algorithm," *Neural Networks*, vol. 2, pp. 475–494.

Peterson, C., and B. Söderberg, 1989. "A new method of mapping optimization problems onto neural networks," *International Journal of Neural Systems*, vol. 1, pp. 3–22.

Pham, D.T., and P. Garrat, 1997. "Blind separation of mixture of independent sources through a quasi-maximum likelihood approach," *IEEE Transactions on Signal Processing*, vol. 45, pp. 1712–1725.

Pham, D.T., P. Garrat, and C. Jutten, 1992. "Separation of a mixture of independent sources through a maximum likelihood approach," *Proceedings of EUSIPCO*, pp. 771–774.

Phillips, D., 1962. "A technique for the numerical solution of certain integral equations of the first kind," *Journal of Association for Computing Machinery*, vol. 9, pp. 84–97.

Pineda, F.J., 1989. "Recurrent backpropagation and the dynamical approach to adaptive neural computation," *Neural Computation*, vol. 1, pp. 161–172.

Pineda, F.J., 1988a. "Generalization of backpropagation to recurrent and higher order neural networks," in *Neural Information Processing Systems*, D.Z. Anderson, ed., pp. 602–611, New York: American Institute of Physics.

Pineda, F.J., 1988b. "Dynamics and architecture in neural computation," *Journal of Complexity*, vol. 4, pp. 216–245.

Pineda, F.J., 1987. "Generalization of back-propagation to recurrent neural networks," *Physical Review Letters*, vol. 59, pp. 2229-2232.

Pitts, W., and W.S. McCulloch, 1947, "How we know universals: The perception of auditory and visual forms," *Bulletin of Mathematical Biophysics*, vol. 9, pp. 127–147.

Plumbley, M.D., and F. Fallside, 1989. "Sensory adaptation: An information-theoretic viewpoint," *International Joint Conference on Neural Networks*, vol. 2, p. 598, Washington, DC.

Plumbley, M.D., and F. Fallside, 1988. "An information-theoretic approach to unsupervised connectionist models," in *Proceedings of the 1988 Connectionist Models Summer School*, D. Touretzky, G. Hinton, and T. Sejnowski, eds., pp. 239–245. San Mateo, CA: Morgan Kaufmann.

Poggio, T., 1990. "A theory of how the brain might work," *Cold Spring Harbor Symposium on Quantitative Biology*, vol. 5, pp. 899–910.

Poggio, T., and D. Beymer, 1996. "Learning to see," *IEEE Spectrum*, vol. 33, no. 5, pp. 60–69.

Poggio, T., and S. Edelman, 1990. "A network that learns to recognize three-dimensional objects," *Nature*, vol. 343, pp. 263–266.

Poggio, T., and F. Girosi, 1990a. "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, pp. 1481–1497.

Poggio, T., and F. Girosi, 1990b. "Regularization algorithms for learning that are equivalent to multilayer networks," *Science*, vol. 247, pp. 978–982.

Poggio, T., and C. Koch, 1985. "Ill-posed problems in early vision: From computational theory to analogue networks," *Proceedings of the Royal Society of London*, Series B, vol. 226, pp. 303–323.

Poggio, T., V. Torre, and C. Koch, 1985. "Computational vision and regularization theory," *Nature*, vol. 317, pp. 314–319.

Polak, E., and G. Ribiére, 1969. "Note sur la convergence de methods de directions conjuguees," *Revue Française Information Recherche Operationnelle* vol. 16, pp. 35–43.

Pöppel G., and U. Krey, 1987. "Dynamical learning process for recognition of correlated patterns in symmetric spin glass models," *Europhysics Letters*, vol. 4, pp. 979–985.

Powell, M.J.D., 1992. "The theory of radial basis function approximation in 1990," in W. Light, ed., *Advances in Numerical Analysis Vol. II: Wavelets, Subdivision Algorithms, and Radial Basis Functions*, pp. 105–210, Oxford: Oxford Science Publications.

Powell, M.J.D., 1988. "Radial basis function approximations to polynomials," *Numerical Analysis 1987 Proceedings*, pp. 223–241, Dundee, UK.

Powell, M.J.D., 1985. "Radial basis functions for multivariable interpolation: A review," *IMA Conference on Algorithms for the Approximation of Functions and Data*, pp. 143–167, RMCS, Shrivenham, England.

Powell, M.J.D., 1977. "Restart procedures for the conjugate gradient method," *Mathematical Programming*, vol. 12, pp. 241–254.

Preisendorfer, R.W., 1988. *Principal Component Analysis in Meteorology and Oceanography*, New York: Elsevier.

Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, 1988. *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge: Cambridge University Press.

Proakis, J.G., 1989. *Digital Communications*, 2nd edition, New York: McGraw-Hill.

Prokhorov, D.V., and D.C. Wunsch, II, 1997. "Adaptive critic designs," *IEEE Transactions on Neural Networks*, vol. 8, pp. 997–1007.

Puskorius, G.V., and L.A. Feldkamp, 1994. "Neurocontrol of nonlinear dynamical systems with Kalman filter-trained recurrent networks," *IEEE Transactions on Neural Networks*, vol. 5, pp. 279–297.

Puskorius, G.V., and L.A. Feldkamp, 1992. "Model reference adaptive control with recurrent networks trained by the dynamic DEKF algorithm," *International Joint Conference on Neural Networks*, vol. II, pp. 106–113, Baltimore.

Puskorius, G.V., L.A. Feldkamp, and L.I. Davis, Jr., 1996. "Dynamic neural network methods applied to on-vehicle idle speed control," *Proceedings of the IEEE*, vol. 84, pp. 1407–1420.

Puskorius, G.V., and L.A. Feldkamp, 1991. "Decoupled extended Kalman filter training of feedforward layered networks," *International Joint Conference on Neural Networks*, vol. 1, pp. 771–777, Seattle.

Rabiner, L.R., 1989. "A tutorial on hidden Markov models," *Proceedings of the IEEE*, vol. 73, pp. 1349–1387.

Rabiner, L.R., and B.H. Juang, 1986. "An introduction to hidden Markkov models," *IEEE ASSP Magazine*, vol. 3, pp. 4–16.

Rall, W., 1989. "Cable theory for dendritic neurons," in *Methods in Neuronal Modeling*, C. Koch and I. Segev, eds., pp. 9–62, Cambridge, MA: MIT Press.

Rall, W., 1990. "Some historical notes," in *Computational Neuroscience*, E.L. Schwartz, Ed., pp. 3–8, Cambridge: MIT Press.

Ramón y Cajál, S., 1911, *Histologie du Systéms Nerveux de l'homme et des vertébrés*, Paris: Maloine.

Rao, A., D. Miller, K. Rose, and A. Gersho, 1997a. "Mixture of experts regression modeling by deterministic annealing," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2811–2820.

Rao, A., K. Rose, and A. Gersho, 1997b. "A deterministic annealing approach to discriminative hidden Markov model design," *Neural Networks for Signal Processing VII, Proceedings of the 1997 IEEE Workshop*, pp. 266–275, Amelia Island, FL.

Rao, C.R., 1973. *Linear Statistical Inference and Its Applications*, New York: Wiley.

Rashevsky, N., 1938. *Mathematical Biophysics*, Chicago: University of Chicago Press.

Raviv, Y., and N. Intrator, 1996. "Bootstrapping with noise: An effective regularization technique," *Connection Science*, vol. 8, pp. 355–372.

Reed, R., 1993. "Pruning algorithms—A survey," *IEEE Transactions on Neural Networks*, vol. 4, pp. 740–747.

Reeke, G.N. Jr., L.H. Finkel, and G.M. Edelman, 1990. "Selective recognition automata," in *An Introduction to Neural and Electronic Networks*, S.F. Zornetzer, J.L. Davis, and C. Lau, eds,, pp. 203–226, New York: Academic Press.

Reif, 1965. *Fundamentals of Statistical and Thermal Physics*, New York: McGraw-Hill.

Renals, S., 1989. "Radial basis function network for speech pattern classification," *Electronics Letters*, vol. 25, pp. 437–439.

Rényi, A. 1960. "On measures of entropy and information," *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics, and Probability*, pp. 547–561.

Rényi, A., 1970. *Probability Theory*, North-Holland, Amsterdam.

Richard, M.D., and R.P. Lippmann, 1991. "Neural network classifiers estimate Bayesian a posteriori probabilities," *Neural Computation*, vol. 3, pp. 461–483.

Riesz, F., and B. Sz-Nagy, 1955. *Functional Analysis*, 2nd edition, New York: Frederick Ungar.

Ripley, B.D., 1996. *Pattern Recognition and Neural Networks*, Cambridge: Cambridge University Press.

Rissanen, J., 1978. "Modeling by shortest data description," *Automatica*, vol. 14, pp. 465–471.

Rissanen, J., 1989. *Stochastic Complexity in Statistical Inquiry*, Singapore: World Scientific.

Ritter, H., 1991. "Asymptotic level density for a class of vector quantization processes," *IEEE Transactions on Neural Networks*, vol. 2, pp. 173–175.

Ritter, H., 1995. "Self-organizing feature maps: Kohonen maps," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, pp. 846–851, Cambridge, MA: MIT Press.

Ritter, H., and T. Kohonen, 1989. "Self-organizing semantic maps," *Biological Cybernetics*, vol. 61, pp. 241 254.

Ritter, H., and K. Schulten, 1988. "Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability, and dimension selection," *Biological Cybernetics*, vol. 60, pp. 59–71.

Ritter, H., T.M. Martinetz, and K.J. Schulten, 1989. "Topology-conserving maps for learning visuo-motor-coordination," *Neural Networks*, vol. 2, pp. 159–168.

Ritter, H., T. Martinetz, and K. Schulten, 1992. *Neural Computation and Self-Organizing Maps: An Introduction*, Reading, MA: Addison-Wesley.

Robbins, H., and S. Monro, 1951. "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407.

Robinson, D.A., 1992. "Signal processing by neural networks in the control of eye movements," *Computational Neuroscience Symposium*, pp. 73–78, Indiana University-Purdue University at Indianapolis.

Rochester, N., J.H. Holland, L.H. Haibt, and W.L. Duda, 1956. "Tests on a cell assembly theory of the action of the brain, using a large digital computer," *IRE Transactions on Information Theory*, vol. IT-2, pp. 80–93.

Rose, K., 1998. "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, to appear.

Rose, K., 1991. *Deterministic Annealing, Clustering, and Optimization*, Ph.D. Thesis, California Institute of Technology, Pasadena, CA.

Rose, K., E. Gurewitz, and G.C. Fox, 1992. "Vector quantization by deterministic annealing," *IEEE Transactions on Information Theory*, vol. 38, pp. 1249–1257.

Rose, K., E. Gurewitz, and G.C. Fox, 1990. "Statistical mechanics and phase transitions in clustering," *Physical Review Letters*, vol. 65, pp. 945–948.

Rosenblatt, F., 1962. *Principles of Neurodynamics*, Washington, DC: Spartan Books.

Rosenblatt, F., 1960a. "Perceptron simulation experiments," *Proceedings of the Institute of Radio Engineers*, vol. 48, pp. 301–309.

Rosenblatt, F., 1960b. "On the convergence of reinforcement procedures in simple perceptrons," Cornell Aeronautical Laboratory Report, VG-1196-G-4, Buffalo, NY.

Rosenblatt, F., 1958. "The Perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, pp. 386–408.

Rosenblatt, M., 1970. "Density estimates and Markov sequences," in M. Puri, ed., *Nonparametric Techniques in Statistical Inference*, pp. 199–213, London: Cambridge University Press.

Rosenblatt, M., 1956. "Remarks on some nonparametric estimates of a density function," *Annals of Mathematical Statistics*, vol. 27, pp. 832–837.

Ross, S.M., 1983. *Introduction to Stochastic Dynamic Programming*, New York: Academic Press.

Roth, Z., and Y. Baram, 1996. "Multi-dimensional density shaping by sigmoids," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1291–1298.

Roussas, G., ed., 1991. *Nonparametric Functional Estimation and Related Topics*, The Netherlands: Kluwer.

Roy, S., and J.J. Shynk, 1990. "Analysis of the momentum LMS algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-38, pp. 2088–2098.

Rubner, J., and K. Schulten, 1990. "Development of feature detectors by self-organization," *Biological Cybernetics*, vol. 62, pp. 193–199.

Rubner, J., and P. Tavan, 1989. "A self-organizing network for principal component analysis," *Europhysics Letters*, vol. 10, pp. 693–698.

Rueckl, J.G., K.R. Cave, and S.M. Kosslyn, 1989. "Why are 'what' and 'where' processed by separate cortical visual systems? A computational investigation," *J. Cognitive Neuroscience*, vol. 1, pp. 171–186.

Rumelhart, D.E., and J.L. McClelland, eds., 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, Cambridge, MA: MIT Press.

Rumelhart, D.E., and D. Zipser, 1985. "Feature discovery by competitive learning," *Cognitive Science*, vol. 9, pp.75–112.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams, 1986a. "Learning representations of back-propagation errors," *Nature (London)*, vol. 323, pp. 533–536.

Rumelhart, D.E., G. E. Hinton, and R.J. Williams, 1986b. "Learning internal representations by error propagation," in D.E. Rumelhart and J.L. McCleland, eds., vol 1, Chapter 8, Cambridge, MA: MIT Press.

Russell, S.J., and P. Novig, 1995. *Artificial Intelligence: A Modern Approach*, Upper Saddle River, NJ: Prentice-Hall.

Russo, A.P., 1991. *Neural Networks for Sonar Signal Processing*, Tutorial No. 8, *IEEE Conference on Neural Networks for Ocean Engineering*, Washington, DC.

Ruyck, D.W., S.K. Rogers, M. Kabrisky, M.E. Oxley, and B.W. Suter, 1990. "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Transactions of Neural Networks*, vol. 1, pp. 296–298.

Saarinen, S., R.B. Bramley, and G. Cybenko, 1992. "Neural networks, backpropagation, and automatic differentiation," in *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, A. Griewank and G.F. Corliss, eds., pp. 31–42, Philadelphia: SIAM.

Saarinen, S., R. Bramley, and G. Cybenko, 1991. "The numerical solution of neural network training problems," *CRSD Report No. 1089*, Center for Supercomputing Research and Development, University of Illinois, Urbana, IL.

Säckinger, E., B.E. Boser, J. Bromley, Y. LeCun, and L.D. Jackel, 1992a. "Application of the ANNA neural network chip to high-speed character recognition," *IEEE Transactions on Neural Networks*, vol. 3, pp.498–505.

Säckinger, E., B.E. Boser, and L.D. Jackel, 1992b. "A neurocomputer board based on the ANNA neural network chip," *Advances in Neural Information Processing Systems*, vol. 4, pp. 773–780, San Mateo, CA: Morgan Kaufmann.

Saerens, M., and A. Soquet, 1991. "Neural controller based on back-propagation algorithm," *IEE Proceedings (London), Part F*, vol. 138, pp. 55–62.

Sage, A.P., ed., 1990. *Concise Encyclopedia of Information Processing in Systems and Organizations*, New York: Pergamon.

Salomon, R., and J.L. van Hemmen, 1996. "Accelerating backpropagation through dynamic self-adaptation," *Neural Networks*, vol. 9, pp. 589–601.

Samuel, A.L., 1959. "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 3, pp. 211–229.

Sandberg, I.W., 1991. "Structure theorems for nonlinear systems," *Multidimensional Systems and Signal Processing*, vol. 2, pp. 267–286.

Sandberg, I.W., L. Xu, 1997a. "Uniform approximation of multidimensional myopic maps," *IEEE*

Transactions on Circuits and Systems, vol. 44, pp. 477–485.

Sandberg, I.W., and L. Xu, 1997b. "Uniform approximation and gamma networks," *Neural Networks,* vol. 10, pp. 781–784.

Sanger, T.D., 1990. "Analysis of the two-dimensional receptive fields learned by the Hebbian algorithm in response to random input," *Biological Cybernetics,* vol. 63, pp. 221–228.

Sanger, T.D., 1989a. "An optimality principle for unsupervised learning," *Advances in Neural Information Processing Systems,* vol. 1, pp. 11–19, San Mateo, CA: Morgan Kaufmann.

Sanger, T.D., 1989b. "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks,* vol. 12, pp. 459–473.

Sanner, R.M., and J.-J.E. Slotine, 1992. "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks,* vol. 3, pp. 837–863.

Sauer, N., 1972. "On the densities of families of sets," *Journal of Combinatorial Theory, Series A,* vol. 13, pp. 145–172.

Sauer, T., J.A. Yorke, and M. Casdagli, 1991. "Embedology," *Journal of Statistical Physics,* vol. 65, pp. 579–617.

Saul, L.K., T. Jakkolla, and M.I. Jordan, 1996. "Mean field theory for sigmoid belief networks," *Journal of Artificial Intelligence Research,* vol. 4, pp. 61–76.

Saul, L.K., and M.I. Jordan, 1996. "Exploiting tractable substructures in intractable networks," *Advances in Neural Information Processing Systems,* vol. 8, pp. 486–492, Cambridge, MA: MIT Press.

Saul, L.K., and M.I. Jordan, 1995. "Boltzmann chains and hidden Markov models," *Advances in Neural Information Processing Systems,* vol. 7, pp. 435–442.

Schapire, R.E., 1997. "Using output codes to boost multiclass learning problems," *Machine Learning: Proceedings of the Fourteenth International Conference,* Nashville, TN.

Schapire, R.E., 1990. "The strength of weak learnability," *Machine Learning,* vol. 5, pp. 197–227.

Schapire R.E., Y. Freund, and P. Bartlett, 1997. "Boosting the margin: A new explanation for the effectiveness of voting methods," *Machine Learning: Proceedings of the Fourteenth International Conference,* Nashville, TN.

Schiffman, W.H., and H.W. Geffers, 1993. "Adaptive control of dynamic systems by back propagation networks," *Neural Networks,* vol. 6, pp. 517–524.

Schneider, C.R., and H.C. Card, 1998. "Analog hardware implementation issues in deterministic Boltzmann machines," *IEEE Transactions on Circuits and Systems II,* vol. 45, to appear.

Schneider, C.R., and H.C. Card, 1993. "Analog CMOS deterministic Boltzmann circuits," *IEEE Journal Solid-State Circuits,* vol. 28, pp. 907–914.

Schölkopf, B., 1997. *Support Vector Learning,* Munich, Germany: R. Oldenbourg Verlag.

Schölkopf, B., P. Simard, V. Vapnik, and A.J. Smola, 1997. "Improving the accuracy and speed of support vector machines," *Advances in Neural Information Processing Systems,* vol. 9, pp. 375–381.

Schölkopf, B., A. Smola, and K.-R. Müller, 1998. "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation,* vol. 10, to appear.

Schölkopf, B., K.-K Sung, C.J.C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, 1997. "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Transactions on Signal Processing,* vol. 45, pp. 2758–2765.

Schraudolph, N.N., and T.J. Sejnowski, 1996. "Tempering back propagation networks: Not all weights are created equal," *Advances in Neural Information Processing Systems,* vol. 8, pp. 563–569, Cambridge, MA: MIT Press.

Schumaker, L.L., 1981, *Spline Functions: Basic Theory,* New York: Wiley.

Schurmars, D., 1997. "Alternative metrics for maximum margin classification," *NIPS Workshop on Support Vector Machines,* Beckenbridge, CO.

Schuster, H.G., 1988. *Deterministic Chaos: An Introduction,* Weinheim, Germany: VCH.

Scofield, C.L., and L.N. Cooper, 1985. "Development and properties of neural networks," *Contemporary Physics,* vol. 26, pp. 125–145.

Scott, A.C., 1977. *Neurophysics,* New York: Wiley.

Segee, B.E., and M.J. Carter, 1991. "Fault tolerance of pruned multilayer networks," *International Joint Conference on Neural Networks,* vol. II, pp. 447–452, Seattle.

Sejnowski, T.J., 1977a. "Strong covariance with nonlinearly interacting neurons," *Journal of Mathematical Biology,* vol. 4, pp. 303–321.

Sejnowski, T.J., 1977b. "Statistical constraints on synaptic plasticity," *Journal of Theoretical Biology,* vol. 69, pp. 385–389.

Sejnowski, T.J., 1976. "On global properties of neuronal interaction," *Biological Cybernetics,* vol. 22, pp. 85–95.

Sejnowski, T.J., and P.S. Churchland, 1989. "Brain and cognition," in *Foundations of Cognitive Science,* M.I. Posner, ed., pp. 301–356, Cambridge, MA: MIT Press.

Sejnowski, T.J., P.K. Kienker, and G.E. Hinton, 1986. "Learning symmetry groups with hidden units: Beyond the perceptron," *Physica,* vol. 22D, pp. 260–275.

Sejnowski, T.J., C. Koch, and P.S. Churchland, 1988. "Computational neuroscience," *Science*, vol. 241, pp. 1299–1306.

Sejnowski, T.J., and C.R. Rosenberg, 1987. "Parallel networks that learn to pronounce English text," *Complex Systems*, vol. 1, pp. 145–168.

Sejnowski, T.J., B.P. Yuhas, M.H. Goldstein, Jr., and R.E. Jenkins, 1990. "Combining visual and acoustic speech signals with a neural network improves intelligibility." *Advances in Neural Information Processing Systems*, vol. 2, pp. 232–239, San Mateo, CA: Morgan Kaufmann.

Selfridge, O.G., R.S. Sutton, and C.W. Anderson, 1988. "Selected bibliography on connectionism," *Evolution, Learning, and Cognition*, Y.C. Lee, Ed., pp. 391–403, River Edge, NJ: World Scientific Publishing, Inc.

Seung, H., 1995. "Annealed theories of learning," in J.-H Oh, C. Kwon, and S. Cho, eds., *Neural Networks. The Statistical Mechanics Perspective*, Singapore: World Scientific.

Seung, H.S., T.J. Richardson, J.C. Lagarias, and J.J. Hopfield, 1998. "Saddle point and Hamiltonian structure in excitatory-inhibitory networks," *Advances in Neural Information Processing Systems*, vol. 10, to appear.

Shah, S., and F. Palmieri, 1990. "MEKA—A fast, local algorithm for training feedforward neural networks," *International Joint Conference on Neural Networks*, vol.3, pp. 41–46, San Diego, CA.

Shamma, S., 1989. "Spatial and temporal processing in central auditory networks," in *Methods in Neural Modeling*, C. Koch and I. Segev, Eds., Cambridge, MA: MIT Press.

Shanno, D.F., 1978. "Conjugate gradient methods with inexact line searches," *Mathematics of Operations Research*, vol. 3, pp. 244–256.

Shannon, C.E., 1948. "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656.

Shannon, C.E., and W. Weaver, 1949. *The Mathematical Theory of Communication*, Urbana, IL.: The University of Illinois Press.

Shannon, C.E., and J. McCarthy, eds., 1956. *Automata Studies*, Princeton, NJ: Princeton University Press.

Shepherd, G.M., 1988. *Neurobiology*, 2nd edition, New York: Oxford University Press.

Shepherd, G.M., 1978. "Microcircuits in the nervous system," *Scientific American*, vol. 238, pp. 92–103.

Shepherd, G.M., ed., 1990a. *The Synaptic Organization of the Brain*, 3rd edition, New York: Oxford University Press.

Shepherd, G.M., 1990b. "The significance of real neuron architectures for neural network simulations," in *Computational Neuroscience*, E.L. Schwartz, ed., pp. 82–96, Cambridge: MIT Press.

Shepherd, G.M., and C. Koch, 1990. "Introduction to synaptic circuits," in *The Synaptic Organization of the Brain*, G.M. Shepherd, ed., pp. 3–31. New York: Oxford University Press.

Sherrington, C.S., 1906. *The Integrative Action of the Nervous System*, New York: Oxford University Press.

Sherrington, C.S., 1933. *The Brain and Its Mechanism*, London: Cambridge University Press.

Sherrington, D., and S. Kirkpatrick, 1975. "Spin-glasses," *Physical Review Letters*, vol. 35, p. 1972.

Shewchuk, J.R., 1994. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, August 4, 1994.

Shore, J.E., and R.W. Johnson, 1980. "Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy," *IEEE Transactions on Information Theory*, vol. IT-26, pp. 26–37.

Shynk, J.J., 1990. "Performance surfaces of a single-layer perceptron," *IEEE Transactions or Neural Networks*, 1, 268–274.

Shynk, J.J. and N.J. Bershad, 1991. Steady-state analysis of a single-layer perceptron based on a system identification model with bia terms," *IEEE Transactions on Circuits and Systems*, vol. CAS-38, pp. 1030–1042.

Shustorovich, A., 1994. "A subspace projection approach to feature extraction: The two-dimensional Gabor transform for character recognition," *Neural Networks*, vol. 7, pp. 1295–1301.

Shustorovich, A., and C. Thrasher, 1996. "Neural network positioning and classification of handwritten characters," *Neural Networks*, vol. 9, pp. 685–693.

Shynk, J.J., and N.J. Bershad, 1992. "Stationary points and performance surfaces of a perceptron learning algorithm for a nonstationary data model," *International Joint Conference on Neural Networks*, vol. 2, pp. 133–139, Baltimore.

Shynk, J.J., and N.J. Bershad, 1991. "Steady-state analysis of a single-layer perceptron based on a system identification model with bias terms," *IEEE Transactions on Circuits and Systems*, vol. CAS-38, pp. 1030–1042.

Siegelmann, H.T., B.G. Horne, and C.L. Giles, 1997 "Computational capabilities of recurrent NARX neural networks," *Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 27, pp. 208–215.

Siegelmann, H.T., and E.D. Sontag, 1991. "Turing computability with neural nets," *Applied Mathematics Letters*, vol. 4, pp. 77–80.

Simard, P., Y. LeCun, and J. Denker, 1993. "Efficient pattern recognition using a new transformation distance," *Advances in Neural Information Processing Systems*, vol. 5, pp. 50–58, San Mateo, CA: Morgan Kaufmann.

Simard, P., B. Victorri, Y. LeCun, and J. Denker, 1992. "Tangent prop — A formalism for specifying selected invariances in an adaptive network," *Advances in Neural Information Systems*, vol. 4, pp. 895–903, San

Mateo, CA: Morgan Kaufmann.

Simmons, J.A. 1989. "A view of the world through the bat's ear: The formation of acoustic images in echolocation," *Cognition*, vol. 33, pp. 155–199.

Simmons, J.A., P.A. Saillant, and S.P. Dear, 1992. "Through a bat's ear," *IEEE Spectrum*, vol. 29(3), pp. 46–48.

Singh, S.P., ed., 1992. *Approximation Theory, Spline Functions and Applications*, Dordrecht, The Netherlands: Kluwer.

Singh, S., and D. Bertsekas, 1997. "Reinforcement learning for dynamic channel allocation in cellular telephone systems," *Advances in Neural Information Processing Systems*, vol. 9, pp. 974-980, Cambridge, MA: MIT Press.

Singhal, S., and L. Wu, 1989. "Training feed-forward networks with the extended Kalman filter," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1187–1190, Glasgow, Scotland.

Singleton, R.C., 1962. "A test for linear separability as applied to self-organizing machines," in M.C. Yovitz, G.T. Jacobi, and G.D. Goldstein, eds., *Self Organizing Systems*, pp. 503–524, Washington DC: Spartan Books.

Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky, 1995. "Nonlinear black-box modeling in system identification: A unified overview," *Automatica*, vol. 31, pp. 1691–1724.

Slepian, D., 1973. *Key papers in the development of information theory*, New York: IEEE Press.

Sloane, N.J.A., and A.D. Wyner, 1993. *Claude Shannon: Collected Papers*, New York: IEEE Press.

Slotine, J.-J. and W. Li, 1991. Applied Nonlinear Control, Englewood Cliffs, NJ: Prentice-Hall.

Smith, M., 1993. *Neural Networks for Statistical Modeling*, New York: Van Nostrand Reinhold.

Smola, A.J., and B. Schölkopf, 1998. "From regularization operators to support vector kernels," *Advances in Neural Information Processing Systems*, vol. 10, to appear.

Smolensky, P., 1988. "On the proper treatment of connectionism," *Behavioral and Brain Sciences*, vol. 11, pp. 1–74.

Sontag, E.D., 1996. "Recurrent neural networks: Some learning and systems-theoretic aspects," Department of Mathematics, Rutgers University, New Brunswick, NJ.

Sontag, E.D., 1992. "Feedback stabilization using two-hidden-layer nets," *IEEE Transactions on Neural Networks*, vol. 3, pp. 981–990.

Sontag, E.D., 1990. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, New York: Springer-Verlag.

Sontag, E.D., 1989. "Sigmoids distinguish more efficiently than Heavisides," *Neural Computation*, vol. 1, pp. 470–472.

Southwell, R.V., 1946. *Relaxation Methods in Theoretical Physics*, New York: Oxford University Press.

Specht, D.F., 1991. "A general regression neural network," *IEEE Transactions on Neural Networks*, vol. 2, pp. 568–576.

Sperduti, A., 1997. "On the computational power of recurrent neural networks for structures," *Neural Networks*, vol. 10, pp. 395–400.

Sperduti, A., and A. Starita, 1997. "Supervised neural networks for the classification of structures," *IEEE Transactions on Neural Networks*, vol. 8, pp. 714–735.

Sprecher, D.A., 1965. "On the structure of continuous functions of several variables," *Transactions of the American Mathematical Society*, vol. 115, pp. 340–355.

Steinbuch, K., 1961. "Die Lernmatrix." *Kybernetik*, vol. 1, pp.36–45.

Stent, G.S., 1973. "A physiological mechanism for Hebb's postulate of learning," *Proceedings of the National Academy of Sciences, USA*, vol. 70, pp. 997–1001.

Sterling, P., 1990. "Retina," in *The Synaptic Organization of the Brain*, G.M. Shepherd, ed., 3rd edition, pp. 170–213, New York: Oxford University Press.

Stevenson, M., R. Winter, and B. Widrow, 1990. "Sensitivity of layered neural networks to errors in the weights," *International Joint Conference on Neural Networks*, vol. 1, pp. 337–340, Washington, DC.

Stone, M., 1978. "Cross-validation: A review," *Mathematische Operationsforschung Statistischen, Serie Statistics*, vol. 9, pp. 127–139.

Stone, M., 1974. "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society*, vol. B36, pp. 111–133.

Stork, D., 1989. "Is backpropagation biologically plausible?" *International Joint Conference on Neural Networks*, vol. 2, pp. 241–246, Washington, DC.

Strang, G., 1980. *Linear Algebra and its Applications*, New York: Academic Press.

Stuart, A., and K. Ord, 1994. *Kendall's Advanced Theory of Statistics*, vol. I, 6th edition, New York: Halsted Press.

Su, H.-T., and T. McAvoy, 1991. "Identification of chemical processes using recurrent networks," *Proceedings of the 10th American Controls Conference*, vol. 3, pp. 2314–2319, Boston.

Su, H.-T., T. McAvoy, and P. Werbos, 1992. "Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach," *Industrial Engineering and Chemical Research*, vol. 31,

pp. 1338–1352.

Suga, N., 1990a. "Cortical computational maps for auditory imaging," *Neural Networks*, vol. 3, pp. 3–21.

Suga, N., 1990b. "Computations of velocity and range in the bat auditory system for echo location," in *Computational Neuroscience*, E.L. Schwartz, ed., pp. 213–231, Cambridge, MA: MIT Press.

Suga, N., 1990c. "Biosonar and neural computation in bats," *Scientific American*, vol. 262, pp. 60–68.

Suga, N., 1985. "The extent to which bisonar information is represented in the bat auditory cortex," in *Dynamic Aspects of Neocortical Function*, G.M. Edelman, W.E. Gall, and W.M. Cowan, eds. pp. 653–695, New York: Wiley (Interscience).

Suga, N., and J.S. Kanwal, 1995. "Echolocation: Creating computational maps," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, Cambridge, MA: MIT Press.

Sutton, J.P., and J.A. Anderson, 1995. "Computational and neurobiological features of a network of networks," in J.M. Bower, ed., *The Neurobiology of Computation*, pp. 317–322, Boston: Kluwer.

Sutton, R.S., 1988. "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, pp. 9–44.

Sutton, R.S., 1986. "Two problems with back-propagation and other steepest-descent learning procedures for networks," *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pp. 823–831. Hillsdale, NJ: Lawrence Erlbaum.

Sutton, R.S., ed., 1992. Special Issue on Reinforcement Learning, *Machine Learning*, vol. 8, pp. 1–395.

Sutton, R.S., 1984. "Temporal credit assignment in reinforcement learning," Ph.D. Dissertation, University of Massachusetts, Amherst, MA.

Sutton, R.S., and A.G. Barto, 1998. *Reinforcement Learning: An Introduction*, Cambridge, MA: MIT Press.

Suykens, J.A.K., J.P.L. Vandewalle, and B.L.R. DeMoor, 1996. *Artificial Neural Networks for Modeling and Control of Non-Linear Systems*, Dordrecht, The Netherlands: Kluwer.

Swindlehurst, A.L., M.J. Goris, and B. Ottersten, 1997. "Some experiments with array data collected in actual urban and suburban environments," *IEEE Workshop on Signal Processing Advances in Wireless Communications*, pp. 301–304, Paris, France.

Takahashi, Y., 1993. "Generalization and approximation capabilities of multilayer networks," *Neural Computation*, vol. 5, pp. 132–139.

Takens, F., 1981. "On the numerical determination of the dimension of an attractor," in D. Rand and L.S. Young, eds., *Dynamical Systems and Turbulence*, Annual Notes in Mathematics, vol. 898, pp. 366–381, Berlin: Springer-Verlag.

Tapia, R.A., and J.R. Thompson, 1978. *Nonparametric Probability Density Estimation*, Baltimore: The Johns Hopkins University Press.

Taylor, J.G., 1997. "Neural computation: The historical background," in E. Fiesler and R. Beale, eds., *Handbook of Neural Computation*, New York: Oxford University Press.

Taylor, W.K., 1964. "Cortico-thalamic organization and memory," *Proceedings of the Royal Society, London, Series B*, vol. 159, pp. 466–478.

Taylor, W.K. 1956. "Electrical simulation of some nervous system functional activities," *Information Theory*, vol. 3, E.C. Cherry, ed., pp. 314–328, London: Butterworths.

Tesauro, G., 1995. "Temporal difference learning and TD-gamma," *Communications of the Association for Computing Machinery*, vol. 38, pp. 58–68.

Tesauro, G., 1994. "TD-Gammon, A self-teaching Backgammon program, achieves master-level play," *Neural Computation*, vol. 6, pp. 215–219.

Tesauro, G., 1992. "Practical issues in temporal difference learning," *Machine Learning*, vol. 8, pp. 257–277.

Tesauro, G., 1989. "Neurogammon wins computer olympiad," *Neural Computation*, vol. 1, pp. 321–323.

Tesauro, G., and T.J. Sejnowski, 1989. "A parallel network that learns to play backgamma," *Artificial Intelligence*, vol. 39, pp. 357–390.

Tesauro, G., and R. Janssens, 1988. "Scaling relationships in back-propagation learning," *Complex Systems*, vol. 2, pp. 39–44.

Teyler, T.J., 1986. "Memory: Electrophysiological analogs," in *Learning and Memory: A Biological View*, J.L. Martinez, Jr. and R.S. Kesner, eds., pp. 237–265, New York: Academic Press.

Thorndike, E.L., 1911. *Animal Intelligence*, Darien, CT: Hafner.

Thrun, S.B., 1992. "The role of exploration in learning control," in *Handbook of Intelligent Control*, D.A. White and D.A. Sofge, eds., pp. 527–559, New York: Van Nostrand Reinhold.

Tikhonov, A.N., 1973. "On regularization of ill-posed problems," *Doklady Akademii Nauk USSR*, vol. 153, pp. 49–52.

Tikhonov, A.N., 1963. "On solving incorrectly posed problems and method of regularization," *Doklady Akademii Nauk USSR*, vol. 151, pp. 501–504.

Tikhonov, A.N., and V.Y. Arsenin, 1977. *Solutions of Ill-posed Problems*, Washington, DC: W.H. Winston.

Titterington, D.M., A.F.M. Smith, and V.E. Makov, 1985. *Statistical Analysis of Finite Mixture Distributions*,

New York: Wiley.

Touretzky, D.S., and D.A. Pomerleau, 1989. "What is hidden in the hidden layers?" *Byte*, vol. 14, pp. 227–233.

Tsitsiklis, J.N., 1994. "Asynchronous stochastic approximation and Q-learning," *Machine Learning*, vol. 16, pp. 185–202.

Tsoi, A.C., and A.D. Back, 1994. "Locally recurrent globally feedforward networks: A critical review," *IEEE Transactions on Neural Networks*, vol. 5, pp. 229–239.

Turing, A.M., 1952. "The chemical basis of morphogenesis," *Philosophical Transactions of the Royal Society, B*, vol. 237, pp. 5–72.

Turing, A.M., 1950. "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460.

Turing, A.M., 1936. "On computable numbers with an application to the Entscheidungs problem," *Proceedings of the London Mathematical Society*, Series 2, vol. 42, pp. 230–265. Correction published in vol. 43, pp. 544–546.

Tsoi, A.C., and A. Back, 1994. "Locally recurrent globally feedforward networks: A critical review," *IEEE Transactions on Neural Networks*, vol. 5, pp. 229–239.

Tzefestas, S.G., ed., 1997. *Methods and Applications of Intelligent Control*, Boston: Kluwer.

Udin, S.B., and J.W. Fawcett, 1988. "Formation of topographic maps," *Annual Review of Neuroscience*, vol. 2, pp. 289–327.

Ukrainec, A.M., and S. Haykin, 1996. "A modular neural network for enhancement of cross-polar radar targets," *Neural Networks*, vol. 9, pp. 143–168.

Ukrainec, A., and S. Haykin, 1992. "Enhancement of radar images using mutual information based unsupervised neural networks," *Canadian Conference on Electrical and Computer Engineering*, pp. MA6.9.1–MA6.9.4, Toronto, Canada.

Uttley, A.M., 1979. *Information Transmission in the Nervous System*, London: Academic Press.

Uttley, A.M., 1970. "The informon: A network for adaptive pattern recognition," *Journal of Theoretical Biology*, vol. 27, pp. 31–67.

Uttley, A.M., 1966. "The transmission of information and the effect of local feedback in theoretical and neural networks," *Brain Research*, vol. 102, pp. 23–35.

Uttley, A.M., 1956. "A theory of the mechanism of learning based on the computation of conditional probabilities," *Proceedings of the First International Conference on Cybernetics*, Namur, Gauthier-Villars, Paris.

Vaillant, R., C. Monrocq, and Y. LeCun, 1994. "Original approach for the localization of objects in images," *IEE Proceedings (London) on Vision, Image and Signal Processing*, vol. 141, pp. 245–250.

Valavanis, K.P., and G.N. Saridis, 1992. *Intelligent Robotic Systems: Theory, Design, and Applications*, Norwell, MA: Kluwer.

Valiant, L.G., 1984. "A theory of the learnable," *Communications of the Association for Computing Machinery*, vol. 27, pp. 1134–1142.

Vanderbei, R., 1994. "Interior point methods: Algorithms and formulations," *ORSA Journal on Computing*, vol. 6, pp. 32–34.

Van Essen, D.C., C.H. Anderson, and D.J. Felleman, 1992. "Information processing in the primate visual system: An integrated systems perspective," *Science*, vol. 255, pp. 419–423.

van de Laar, P., T. Heskes, and S. Gielen, 1997. "Task-dependent learning of attention," *Neural Networks*, vol. 10, pp. 981–992.

van Laarhoven, P.J.M., and E.H.L. Aarts, 1988. *Simulated Annealing: Theory and Applications*, Boston: Kluwer Academic Publishers.

Van Trees, H.L., 1968. *Detection, Estimation, and Modulation Theory*, Part I, New York: Wiley.

Van Hulle, M.M., 1997. "Nonparametric density estimation and regression achieved with topographic maps maximizing the information-theoretic entropy of their outputs," *Biological Cybernetics*, vol. 77, pp. 49–61.

Van Hulle, M.M., 1996. "Topographic map formation by maximizing unconditional entropy: A plausible strategy for "on-line" unsupervised competitive learning and nonparametric density estimation," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1299–1305.

Van Veen, B., 1992. "Minimum variance beamforming," in S. Haykin and A. Steinhardt, eds., *Adaptive Radar Detection and Estimation*, New York: Wiley (Interscience).

Vapnik, V.N., 1998. *Statistical Learning Theory*, New York: Wiley.

Vapnik, V.N., 1995. *The Nature of Statistical Learning Theory*, New York: Springer-Verlag.

Vapnik, V.N., 1992. "Principles of risk minimization for learning theory," *Advances in Neural Information Processing Systems*, vol. 4, pp. 831–838, San Mateo, CA: Morgan Kaufmann.

Vapnik, V.N., 1982. *Estimation of Dependences Based on Empirical Data*, New York: Springer-Verlag.

Vapnik, V.N., and A.Ya. Chervonenkis, 1971. "On the uniform convergence of relative frequencies of events to their probabilities," *Theoretical Probability and Its Applications*, vol. 17, pp. 264–280.

Vapnik, V.N., and A. Ya. Chervonenkis, 1964. "A note on a class of perceptrons," *Automation and Remote Control*, vol. 25, pp. 103–109.

Velmans, M., 1995. "Consciousness, Theories of." In M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, pp. 247–250, Cambridge, MA: MIT Press.

Venkataraman, S., 1994. "On encoding nonlinear oscillations in neural networks for locomotion," *Proceedings of the 8th Yale Workshop on Adaptive and Learning Systems*, pp. 14–20, New Haven, CT.

Venkatesh, S.J., G. Panche, D. Psaltis, and G. Sirat, 1990. "Shaping attraction basins in neural networks," *Neural Networks*, vol. 3, pp. 613–623.

Vetterli, M., and J. Kovačević, 1995. *Wavelets and Subband Coding*, Englewood Cliffs, NJ: Prentice-Hall.

Vidyasagar, M., 1997. *A Theory of Learning and Generalization*, London: Springer-Verlag.

Vidyasagar, M., 1993. *Nonlinear Systems Analysis*, 2nd edition, Englewood Cliffs, NJ: Prentice-Hall.

Viterbi, A.J., 1967. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260–269.

von der Malsburg, C., 1990a. "Network self-organization," in *An Introduction to Neural and Electronic Networks*, S.F. Zornetzer, J.L. Davis, and C. Lau, eds., pp. 421–432, San Diego, CA: Academic Press.

von der Malsburg, C., 1990b. "Considerations for a visual architecture," in *Advanced Neural Computers*, R. Eckmiller, ed., pp. 303–312, Amsterdam: North-Holland.

von der Malsburg, C., 1981. "The correlation theory of brain function," *Internal Report 81–2*, Department of Neurobiology, Max-Plak-Institute for Biophysical Chemistry, Göttingen, Germany.

von der Malsburg, C., 1973. "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, pp. 85–100.

von der Malsburg, C., and W. Schneider, 1986. "A neural cocktail party processor," *Biological Cybernetics*, vol. 54, pp. 29–40.

von Neumann, J., 1986. *Papers of John von Neumann on Computing and Computer Theory*, W. Aspray and A. Burks, eds., Cambridge, MA: MIT Press.

von Neumann, J., 1958. *The Computer and the Brain*, New Haven, CT: Yale University Press.

von Neumann, J., 1956. "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies*, C.E. Shannon and J. McCarthy, eds., pp. 43–98, Princeton, NJ: Princeton University Press.

Wahba, G., 1990. *Spline Models for Observational Data*, SIAM.

Wahba, G., D.R. Johnson, F. Gao, and J. Gong, 1995. "Adaptive tuning of numerical weather prediction models: Randomized GCV in three and four dimensional data assimilation," *Monthly Weather Review*, vol. 123, pp. 3358–3369.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, 1989. "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-37, pp. 328–339.

Waltz, D., 1997. "Neural nets and AI: Time for a synthesis," plenary talk, *International Conference on Neural Networks*, vol. 1, p. xiii, Houston.

Waltz, M.D., and K.S. Fu, 1965. "A heuristic approach to reinforcement learning control systems," *IEEE Transactions on Automatic Control*, vol. AC-10, pp. 390–398.

Wan, E.A., 1994. "Time series prediction by using a connectionist network with internal delay lines," in *Time Series Prediction: Forecasting the Future and Understanding the Past*, A.S. Weigend and N.A. Gershenfield, eds., pp. 195–217. Reading, MA: Addison-Wesley.

Wan, E.A., 1990. "Temporal backpropagation for FIR neural networks," *IEEE International Joint Conference on Neural Networks*, vol. I, pp. 575–580, San Diego, CA.

Wan, E.A., and F. Beaufays, 1996. "Diagrammatic derivation of gradient algorithms for neural networks," *Neural Computation*, vol. 8, pp. 182–201.

Watanabe, H., Yamaguchi, and S. Katagiri, 1997. "Discriminative metric design for robust pattern recognition," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2655–2662.

Waterhouse, S., D. MacKay, and A. Robinson, 1996. "Bayesian methods for mixtures of experts," *Advances in Neural Information Processing Systems*, vol. 8, pp. 351–357, Cambridge, MA: MIT Press.

Watkins, C.J.C.H., 1989. *Learning from Delayed Rewards*, Ph.D. Thesis, University of Cambridge, England.

Watkins, C.J.C.H., and P. Dayan, 1992. "Q-learning," *Machine Learning*, vol. 8, pp. 279–292.

Watrous, R.L. 1987. "Learning algorithms for connectionist networks: Applied gradient methods of nonlinear optimization," *First IEEE International Conference on Neural Networks*, vol. 2, pp. 619–627, San Diego, CA.

Watson, G.S., 1964. "Smooth regression analysis," *Sankhyā: The Indian Journal of Statistics, Series A*, vol. 26, pp. 359–372.

Webb, A.R., 1994. "Functional approximation by feed-forward networks: A least-squares approach to generalisation," *IEEE Transactions on Neural Networks*, vol. 5, pp. 480–488.

Webb, A.R., and D. Lowe, 1990. "The optimal internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks*, vol. 3, pp. 367–375.

Weigend, A.S., B. Huberman, and D. Rumelhart, 1990. "Predicting the future: A connectionist approach,"

International Journal of Neural Systems, vol. 3, pp. 193–209.

Weigend, A.S., D.E. Rumelhart, and B.A. Huberman, 1991. "Generalization by weight-elimination with application to forecasting, *Advances in Neural Information Processing Systems,* vol. 3, pp. 875–882, San Mateo, CA: Morgan Kaufmann.

Weigend, A.S., and N.A. Gershenfield, eds., 1994. *Time Series Prediction: Forecasting the Future and Understanding the Past,* vol. 15, Santa Fe Institute Studies in the Sciences of Complexity, Reading, MA: Addison-Wesley.

Weierstrass, 1885. "Uber die analytische Darstellbarkeit sogenannter willkurlicher Funktionen einer reellen veranderlichen," *Sitzungsberichte der Akademie der Wissenschaften, Berlin,* pp. 633–639, 789–905.

Werbos, P.J., 1992. "Neural networks and the human mind: New mathematics fits humanistic insight," *IEEE International Conference on Systems, Man, and Cybernetics,* vol. 1, pp. 78–83, Chicago.

Werbos, P.J., 1990. "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE,* vol. 78, pp. 1550–1560.

Werbos, P.J., 1989. "Backpropagation and neurocontrol: A review and prospectus," *International Joint Conference on Neural Networks,* vol. I, pp. 209–216, Washington, DC.

Werbos, P.J., 1974. "Beyond regression: New tools for prediction and analysis in the behavioral sciences." Ph.D. Thesis, Harvard University, Cambridge, MA.

Wettschereck, D., and T. Dietterich, 1992. "Improving the performance of radial basis function networks by learning center locations," *Advances in Neural Information Processing Systems,* vol. 4, pp. 1133–1140, San Mateo, CA: Morgan Kaufmann.

White, D.A., and D.A. Sofge, eds., 1992. *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches,* New York: Van Nostrand Reinhold.

White, H., 1992. *Artificial Neural Networks: Approximation and Learning Theory,* Cambridge, MA: Blackwell.

White, H., 1990. "Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings," *Neural Networks,* vol. 3, pp. 535–549.

White, H., 1989a. "Learning in artificial neural networks: A statistical perspective," *Neural Computation,* vol. 1, pp. 425–464.

White, H., 1989b. "Some asymptotic results for learning in single hidden-layer feedforward network models," *Journal of the American Statistical Society,* vol. 84, pp. 1003–1013.

Whitney, H., 1936. "Differentiable manifolds," *Annals of Mathematics,* vol. 37, pp. 645–680.

Whittaker, E.T., 1923. "On a new method of graduation," *Proceedings of the Edinburgh Mathematical Society,* vol. 41, pp. 63–75.

Widrow, B., 1962. "Generalization and information storage in networks of adeline 'neurons'," in M.C. Yovitz, G.T. Jacobi, and G.D., Goldstein, eds., *Self-Organizing Systems,* pp. 435–461, Washington, DC: Spartan Books.

Widrow, B., J.M. McCool, M.G. Larimore, and C.R. Johnson, Jr., 1976. "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proceedings of the IEEE,* vol. 64, pp.1151–1162.

Widrow, B., J.R. Glover, Jr., J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeidler, J. Dong, Jr., and R.C.Goodlin, 1975. "Adaptive noise cancelling: Principles and applications," *Proceedings of the IEEE,* vol 63, pp. 1692–1716.

Widrow, B., N.K. Gupta, and S. Maitra, 1973. "Punish/reward: Learning with a critic in adaptive threshold systems," *IEEE Transactions of Systems, Man, and Cybernetics,* vol. SMC-3, pp. 455–465.

Widrow, B., and M.E. Hoff, Jr., 1960. "Adaptive switching circuits," *IRE WESCON Convention Record,* pp. 96–104.

Widrow, B., and M.A. Lehr, 1990. "30 years of adaptive neural networks: Perceptron, madaline, and back-propagation," *Proceedings of the Institute of Electrical and Electronics Engineers,* vol. 78, pp. 1415–1442.

Widrow, B., P.E. Mantey, L.J. Griffiths, and B.B. Goode, 1967. "Adaptive antenna systems," *Proceedings of the IEEE,* vol. 55, pp. 2143–2159.

Widrow, B., and S.D. Stearns, 1985. *Adaptive Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall.

Widrow, B., and E. Walach, 1996. *Adaptive Inverse Control,* Upper Saddle River, NJ: Prentice-Hall.

Wieland, A., and R. Leighton, 1987. "Geometric analysis of neural network capabilities," first *IEEE International Conference on Neural Networks,* vol. III, pp. 385–392, San Diego, CA.

Wiener, N., 1961. *Cybernetics,* 2nd edition New York: Wiley.

Wiener, N., 1958. *Nonlinear Problems in Random Theory,* New York: Wiley.

Wiener, N., 1949. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications,* Cambridge, MA: MIT Press. (This was originally issued as a classified National Defense Research Report, February 1942).

Wiener, N., 1948. *Cybernetics: Or Control and Communication in the Animal and the Machine,* New York: Wiley.

Wilks, S.S., 1962. *Mathematical Statistics,* New York: Wiley.

Williams, R.J., 1992. "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256.

Williams, R.J., 1988. "Toward a theory of reinforcement-learning connectionist systems," *Technical Report NU-CCS-88-3*, College of Computer Science, Northeastern University, Boston.

Williams, R.J., 1985. "Feature discovery through error-correction learning," *Technical Report ICS-8501*, University of California, San Diego, CA.

Williams, R.J., and J. Peng, 1990. "An efficient gradient-based algorithm for on-line training of recurrent network trajectories," *Neural Computation*, vol. 2, pp. 490–501.

Williams, R.J., and D. Zipser, 1995. "Gradient-based learning algorithms for recurrent networks and their computational complexity," in Y. Chauvin and D.E. Rumelhart, eds., *Backpropagation: Theory, Architectures, and Applications*, pp. 433–486, Hillsdale, NJ: Lawrence Erlbaum.

Williams, R.J., and D. Zipser, 1989. "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, pp. 270–280.

Willshaw, D.J., O.P. Buneman, and H.C. Longuet-Higgins, 1969. "Non-holographic associative memory," *Nature (London)*, vol. 222, pp. 960–962.

Willshaw, D.J., and C. von der Malsburg, 1976. "How patterned neural connections can be set up by self-organization," *Proceedings of the Royal Society of London Series B*, vol. 194, pp. 431–445.

Wilson, G.V., and G.S. Pawley, 1988. "On the stability of the travelling salesman problem algorithm of Hopfield and Tank," *Biological Cybernetics*, vol. 58, pp. 63–70.

Wilson, H.R., and J.D. Gowan, 1972. "Excitatory and inhibitory interactions in localized populations of model neurons," *Journal of Biophysics*, vol. 12, pp. 1–24.

Winder, R.O., 1961. "Single stage threshold logic," *Switching Circuit Theory and Logical Design*, AIEE Special Publications, vol. S-134, pp. 321–332.

Winograd, S., and J.D. Cowan, 1963. *Reliable Computation in the Presence of Noise*, Cambridge, MA: MIT Press.

Wolpert, D.H., 1992. "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259.

Wood, N.L., and N. Cowan, 1995. "The cocktail party phenomenon revisited: Attention and memory in the classic selective listening procedure of Cherry (1953)," *Journal of Experimental Psychology: General*, vol. 124, pp. 243–262.

Woods, W.A., 1986. "Important issues in knowledge representation," *Proceedings of the Institute of Electrical and Electronics Engineers*, vol. 74, pp. 1322–1334.

Wu, C.F.J., 1983. "On the convergence properties of the EM algorithm," *Annals of Statistics*, vol. 11, pp. 95–103.

Wylie, C.R., and L.C. Barrett, 1982. *Advanced Engineering Mathematics*, 5th edition, New York: McGraw-Hill.

Xu, L., A. Krzyżak, and A. Yuille, 1994. "On radial basis function nets and kernel regression: Statistical consistency, convergency rates, and receptive field size," *Neural Networks*, vol. 7, pp. 609–628.

Xu, L., E. Oja, and C.Y. Suen, 1992. "Modified Hebbian learning for curve and surface fitting," *Neural Networks*, vol. 5, pp. 441–457.

Yang, H., and S. Amari, 1997. "Adaptive online learning algorithms for blind separation: Maximum entropy and minimum mutual information," *Neural Computation*, vol. 9, pp. 1457–1482.

Yee, P.V., 1998. *Regularized Radial Basis Function Networks: Theory and Applications to Probability Estimation, Classification, and Time Series Prediction*, Ph.D. Thesis, McMaster University, Hamilton, Ontario.

Yockey, H.P., 1992. *Information Theory and Molecular Biology*, Cambridge: Cambridge University Press.

Yoshizawa, S., M. Morita, and S. Amari, 1993. "Capacity of associative memory using a nonmonotonic neuron model," *Neural Networks*, vol. 6, pp. 167–176.

Zadeh, L.A., 1973. "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, pp. 28–44.

Zadeh, L.A., 1965. "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353.

Zadeh, L.A., 1953. "A contribution to the theory of nonlinear systems," *J. Franklin Institute*, vol. 255, pp. 387–401.

Zadeh, L.A., and C.A. Desoer, 1963. *Linear System Theory: The State Space Approach*, New York: McGraw-Hill.

Zames, G., 1981. "Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses," *IEEE Transactions on Automatic Control*, vol. AC-26, pp. 301–320.

Zames, G., and B.A. Francis, 1983. "Feedback, minimax, sensitivity, and optimal robustness," *IEEE Transactions on Automatic Control*, vol. AC-28, pp. 585–601.

Zeevi, A.J., R. Meir, and V. Majorov, 1998. "Error bounds for functional approximation and estimation using mixtures of experts," *IEEE Transactions on Information Theory*, vol. 44, pp. 1010–1025.

Zeki, S., 1993. *A Vision of the Brain*, Oxford: Blackwell Scientific Publications.

Zipser, D., and D.E. Rumelhart, 1990. "The neurobiological significance of the new learning models," in *Computational Neuroscience*, E.L. Schwartz, ed., pp. 192–200, Cambridge, MA: MIT Press.

索　引

索引中的页码为英文原书页码，与书中边栏页码一致。

［Ｇｅｎｅｒａｌ Ｉｎｆｏｒｍａｔｉｏｎ］
书名＝神经网络原理
作者＝
页数＝６３３
ＳＳ号＝０
出版日期＝